REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
07-12-2007	FINAL	21 Apr 2005 - 31 Jan 2007

4. TITLE AND SUBTITLE

An Enhanced Collaborative Software Environment for Information Fusion at the Unit of Action.

5a. CONTRACT NUMBER

W15P7T-05-P621

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Corkill, Daniel

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Massachusetts
140 Governors Drive, CMPSCI Dept.
Amherst MA 01003-9264

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Grant and Contract Administration
70 Butterfield Terrace
University of Massachusetts
Amherst MA 01003

10. SPONSOR/MONITOR'S ACRONYM(S)

UMASS-Amherst

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author, and not necessarily shared by the Army.

14. ABSTRACT

This report describes research associated with the development of CIFA (Collaborative Information-Fusion Assistant), a highly responsive decision-support environment that can improve the effectiveness of analysts and decision makers within the Army's Brigade Combat Teams (BCTs). CIFA helps Army analysts and decision makers in answering Priority Intelligence Requirements (PIRs) by focusing their attention on appropriate data, by providing spatially and temporally aggregated views of the environment, and by ensuring that important information has not been overlooked. Research activities were performed in three main areas: 1) blackboard-based temporal and spatial aggregation and abstraction; 2) presentation of real-time battlespace assessments and user alerts; and 3) principled integration of sensor data, human-generated reports, and automated processing results. This report discusses the issues addressed, the techniques developed and evaluations of them, and lessons learned.

15. SUBJECT TERMS

high-level information fusion, PIR answering, battlefield intelligence assessment, decision-support environment, spatial and temporal aggregation and abstraction, AI blackboard system

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Daniel D. Corkill
U	U	U	SAR	50 (w/cover)	19b. TELEPHONE NUMBER *(Include area code)* 413-545-0675

An Enhanced Collaborative-Software Environment for Information Fusion at the Unit of Action

Final Report

Period of Performance: April 21, 2005–January 31, 2007
Sponsored by: U.S. Army RDECOM CERDEC
Intelligence and Information Warfare Directorate
Contract: W15P7T-05-C-P621

Daniel D. Corkill
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003
corkill@cs.umass.edu.edu

Abstract

This report describes research associated with the development of a highly responsive decision-support technology that can improve the effectiveness of analysts and decision makers within the Army's Brigade Combat Teams (BCTs). These analysts and decision makers must work with large data volumes in time-constrained and uncertain operating environments. This joint research and proof-of-concept effort involved the University of Massachusetts Amherst (UMass), BBTech Corporation, and the U.S. Army RDECOM CERDEC Intelligence and Information Warfare Directorate, Fort Monmouth, NJ. This final report focuses on the UMass portion of the effort.

This research was performed in the context of the CIFA (Collaborative Information-Fusion Assistant) decision-support environment, a prototype suite of tools and technologies developed jointly in this effort. CIFA can augment and support Army personnel in answering Priority Intelligence Requirements (PIRs) associated with monitoring, assessing, and responding to enemy courses of action and other battlespace-environment characteristics. At present, time constraints and information overload often result in hasty, partial analysis of the information available to intelligence personnel. CIFA helps Army analysts and decision makers focus their attention on appropriate data by providing spatially and temporally aggregated views of the environment and by ensuring that important information has not been overlooked.

Research activities were performed in three main areas: 1) blackboard-based temporal and spatial aggregation and abstraction; 2) presentation of real-time battlespace assessments and user alerts; and 3) principled integration of sensor data, human-generated reports, and automated processing results. This report discusses the issues we addressed, the techniques we developed and our evaluations of them, and lessons learned. The report concludes with a summary of remaining technical challenges and recommendations for future research and development activities.

The research reported in this document was performed in connection with contract W15P7T-05-C-P621 under the "Fusion Based Knowledge for the Future Force" ATO program and the "Advanced REsearch Solutions - Fused Intelligence with Speed and Trust" program at the U.S. Army RDECOM CERDEC Intelligence and Information Warfare Directorate, Fort Monmouth, NJ. The views and conclusions contained in this document are those of the author and should not be interpreted as presenting the official policies or position, either expressed or implied, of the University of Massachusetts Amherst, the U.S. Army RDECOM CERDEC Intelligence and Information Warfare Directorate, or the U.S. government unless so designated by other authorized documents. Citation of manufacturers or trade names does not constitute an official endorsement or approval of the use thereof.

Contents

1 Introduction

The overall objective of this research effort is developing the scientific foundation and experience necessary to create a highly responsive information-fusion application that improves the effectiveness of Army analysts and decision makers operating at the brigade (BCT) level. These analysts and decision makers must work with large data volumes in time-constrained and uncertain operating environments.

This research was performed in the context of the CIFA (Collaborative Information-Fusion Assistant) decision-support environment (Figure 1), a prototype suite of tools and technologies developed jointly in this effort by the University of Massachusetts Amherst (UMass), BBTech Corporation,[1] and the U.S. Army RDECOM CERDEC Intelligence and Information Warfare Directorate, Fort Monmouth, NJ. CIFA is designed to augment and support field personnel in answering Priority Intelligence Requirements (PIRs) associated with monitoring, assessing, and responding to enemy courses of action and other battlespace-environment characteristics. A major challenge in CIFA is managing the combinatorial explosion of sensing and processing activities without sacrificing accurate inference. The large volumes of data, possibilities, and outcomes exceed human perceptual and cognitive abilities and require an effective human/computer partnership to make the best use of sensing, computation, and communication resources in highly dynamic and uncertain battlefield environments. At present, time constraints and information overload often result in hasty, partial analysis of the information available to intelligence personnel. An effective, automated decision-support application for information fusion and situation assessment can help Army analysts and decision makers focus their attention on appropriate data by providing spatially and temporally aggregated views of the environment and by ensuring that important information has not been overlooked.

The UMass portion of this research focused on representation strategies for effective reasoning, temporal and spatial aggregation techniques, and control of reasoning processes. As will be discussed, it quickly became apparent that significant collective information is present in the stream of individual human-generated and automated-sensor intelligence, surveillance, and reconnaissance (ISR) reports that are available to analysts and decision makers. Harvesting this rich collective information requires *semantically aggregating individual reports in both space and time.* Rather than discarding detailed spatial and temporal information in order to simplify automated reasoning, we investigated how to make use of all the information that can be obtained from ISR and other sources and how to off load the semantic aggregation and reasoning that is now performed manually to CIFA.

2 The CIFA Decision-Support Environment

The CIFA decision-support environment consists of a number of loosely connected component systems, as shown in Figure 1. UMass Amherst developed the CIFA Reasoning Engine (CIFAR)[2] and the C/JMTK-based "Graphical Situation Presentation" components of the CIFA architecture. These components are shown in yellow the Figure.

As a decision-support system, information flow in CIFA begins and ends with its users. CIFA's primary users are the members of the S2-level intelligence staff. This staff use PIRManager to specify and maintain the active set of Priority Intelligence Requirements (PIR) and Specific Information Requirements (SIR). These PIR/SIR inform the CIFAR engine of the kinds of events and enemy behaviors that are of interest to the S2 staff. Additionally, any required knowledge about the environment, weather, terrain, culture, sensor platforms and intelligence report characteristics, equipment and personnel capabilities, and the composition and structure of friendly and enemy forces is provided to CIFAR through the Knowledge Server. Most of this type of knowledge will be developed and loaded into the Knowledge

[1] BBTech Corporation was supported under separate contracts.

[2] Pronounced "see-far."

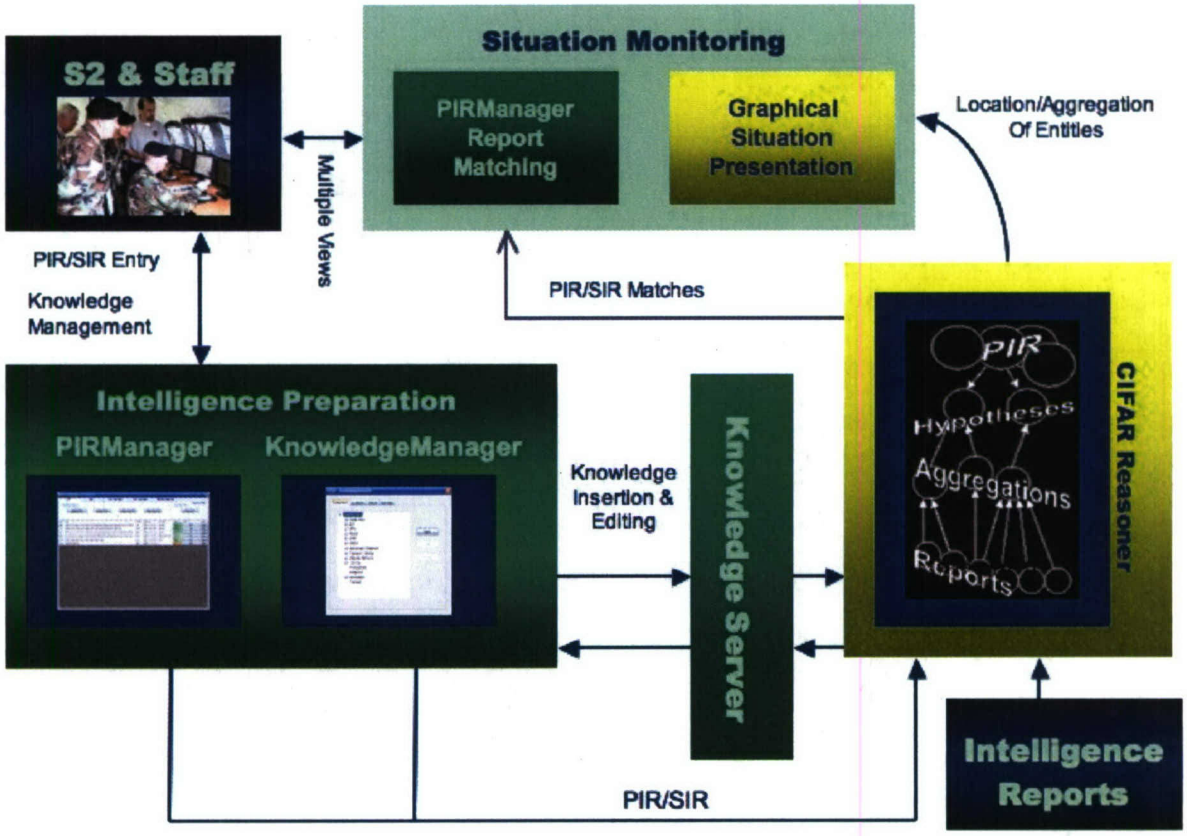


Figure 1: CIFA Architecture

Server in advance of combat operations. If additional "in the field"-specific information is required, the S2 staff can specify it using the KnowledgeManager component.

CIFAR takes as its input:

- knowledge of the environment, reporting, equipment, and force structures from the Knowledge Server
- the PIR and SIR obtained from PIRManager
- the stream of intelligence reports

The CIFAR engine provides decision-support assistance to the S2 staff through two user-interface (UI) components. The first UI component is the CIFA Graphical User Interface (CIFA/UI), which presents an interactive map-based presentation of the hypothesized location, identification, and movement of entities and aggregated groups of entities that have been reported in the environment (Figure 2). The second UI component is the PIR/SIR status monitor (see Figure 22, page 28), which is integrated with PIRManager. The PIR/SIR monitor UI displays current status of all PIR/SIR and notifies the S2 staff of changes in hypothesized entity behaviors in the environment that are relevant to determining the status of a PIR. These two CIFA UI components allow the S2 to quickly understand the situation and focus his or her attention to entity behaviors and their supporting intelligence reports, saving time and making the most effective use of cognitive resources. The presentation views provided by these UI components are individually parametrized and customizable, allowing each user to personalize the tools for the most comfortable and effective results.

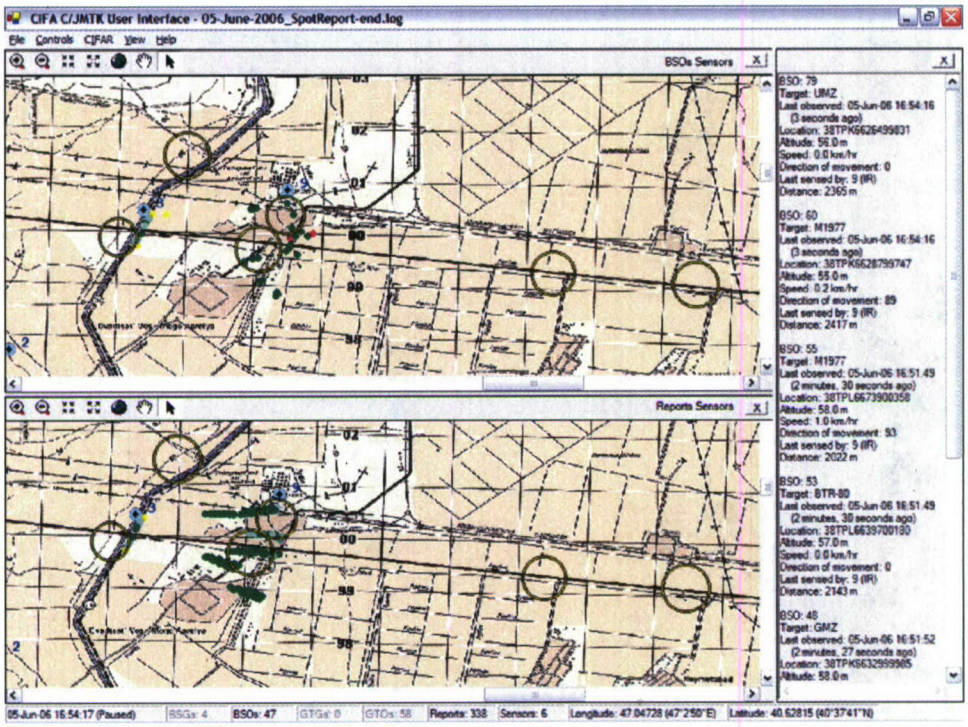


Figure 2: The CIFA Graphical User Interface

3 CIFAR: The CIFA Reasoning Engine

The blackboard-based CIFAR engine performs the inferencing required to identify, aggregate, and track entity activities using the high-volume stream of automated-sensor and human observation reports. CIFAR is implemented using the open-source GBBopen framework.[3] GBBopen is a modern, high-performance, open source blackboard-system development environment that is based on concepts that were explored and refined in the UMass Generic Blackboard system [1] and the commercial GBB product [2]. A major capability provided by GBBopen is multidimensional abstraction over blackboard objects ("unit instances"), blackboard levels/containers ("space instances"), and proximity-based retrieval patterns. Multidimensional abstraction provides a semantically meaningful separation of blackboard-repository storage mechanisms from knowledge source (KS) and control code. This separation allows storage and search strategies and optimizations to change dynamically in order to maintain top performance. GBBopen also provides "link based" inter-object relationships and highly efficient and extensible event signaling/handling that form the foundation for fast, yet flexible, opportunistic processing activities and control reasoning.

GBBopen is written in the Common Lisp language, an ANSI standard.[4] The Common Lisp language is supported by commercial vendors and open-source implementations on a wide range of hardware platforms and operating systems. At the implementation level, GBBopen is designed as a smooth extension of Common Lisp, CLOS (the Common Lisp Object System) [3], and the Meta-object Protocol [4], providing all the advantages of a rich, dynamic, reflective, and extensible language to blackboard-application architects and component writers. These capabilities are crucial in building complex blackboard-based

[3] http://GBBopen.org

[4] Common Lisp was the first ANSI standard to incorporate object-oriented programming: ANSI X3.226-1994.

applications where representations, KSs, and control mechanisms will change during the development and over the operational lifetime of the application.

GBBopen incorporates over 20 years of experience in developing and delivering a wide range of blackboard-system-based applications. In addition to CIFAR, GBBopen is used in many complex and challenging applications including:

- CNAS (AFRL)—power-aware cognitive sensor agents for ground-level environmental monitoring
- INCOMMANDS (DRDC/DND Canada)—anti-missile threat assessment and weapon assignment for Halifax class Canadian Navy frigates
- COORDINATORS (DARPA/CMU team)—agent-based coordination support for humans
- Integrated Learning (DARPA/Lockheed Martin team)—automated learning using integrated learners, reasoners, simulators, planners

3.1 CIFAR Inputs

As highlighted in the CIFA overview, input to CIFAR consists of a stream of individual human and automated sensor reports, the current set of PIR/SIR, and knowledge of sensor and target types and capabilities, activity and behavioral (doctrinal) and strategic knowledge, terrain and weather features, and so on. A key design goal for CIFAR was to encode procedurally as little problem-domain knowledge as possible. Instead, CIFAR obtains this knowledge at start up from the Knowledge Server, and knowledge updates can be loaded dynamically into the operating CIFAR engine at any time.

Intelligence reports Issues of information extraction and data formatting were outside the scope of CIFAR processing. Instead, the intelligence reports input stream consists of well-structured report records containing the information that would realistically be expected to become available in the near term from human and automated intelligence, surveillance, and reconnaissance (ISR) reporting. Major Chet Brown (USAI C&FH) developed the report structure and format, which is detailed in Table 1.

The data sets used in this effort were generated by RDECOM using a suite of tools including JCATS, the Joint Conflict and Tactical Simulation system,[5] and by BBTech Corporation using their MARS (Maneuver And Report-generation Simulator) entity-based modeling and simulation system. The data sets represent realistic report feeds, but they are unclassified and the simulations used to generate them use open source sensor models. Table 2 lists the principal data sets that were used, and we will refer to specific data sets using their abbreviated name (e.g., `ds05`) throughout this report.

In addition to the report data sets, latter JCATS and MARS data sets included a separate input file that contained the ground-truth force structure hierarchy for ground-truth objects (GTOs). As with the GTO labeling of individual reports, this information was used only for evaluation of CIFAR's spatial clustering and was ignored by CIFAR's reasoning activities.

Generating realistic data sets was a difficult task in its own right, and the early JCATS-generated data sets had a number of problems that included incorrect or unavailable target speed and direction values, reporting from sensors that should not be able to see the target, etc. Realistic movement of individual entities was also an issue. If all the tanks move down the road at precisely 40km/hr, it is very easy to determine which future reports are associated with which tank. Adding realistic individual entity variations to JCATS simulations involved labor-intensive "manual animation" to achieve individual behaviors that went beyond random variation from fixed velocity movement. BBTech Corporation's MARS entity-based simulation system was motivated by the need to generate quickly (by automated, entity-based animation) data sets where an individual BSO might need to navigate around terrain or be

[5] http://www.jfcom.mil/about/fact_jcats.htm

Each individual report item in the report stream is formatted as follows:

1. **Time Available** — Military Date/Time Group (DDMonYY, Hour, Minute, Second, Time Zone: `30Dec04,12,00,59,Z`)
2. **Time Sensed** — Military Date/Time Group (DDMonYY, Hour, Minute, Second, Time Zone: `30Dec04,12,00,59,Z`)
3. **Target Type** — Enemy Battle Space Object (BSO) or aggregate type according to unit identification or enemy equipment list codes: `2S3` or `TRACKED`
4. **Target Quantity** — Quantity of BSO or aggregate: `16` (typically `1`)
5. **Target Affiliation**[a] — Reported affiliation of BSO or aggregate: `ENEMY` or `UNKNOWN`
6. **Target Activity** — Description of target activity using Enemy Activity Codes: `MASSG`
7. **Target Direction of Movement** — Cardinal, ordinal, or azimuth (vector) along which the target is moving (field is blank if target is stationary or source cannot provide direction information): `NW` or `270`
8. **Target Speed** — Target velocity in kilometers per hour: `25`
9. **Named Area of Interest (NAI)** — NAI in which target appears if applicable (field is blank if the target is not within an NAI or the source does not provide it: `34`
10. **Target Latitude** — Location of target along the parallel of latitude (North or South, Degrees, Minutes): `N,40,42.033`
11. **Target Longitude** — Location of target along the meridian of longitude (East or West, Degrees, Minutes): `E,47,06.946`
12. **Target Altitude** — Height of target Above Ground Level (AGL) in feet: `6`
13. **Target Military Grid Reference System (MGRS) Location** — Location of target within the MGRS rectangular grid (Grid Zone Designation, 100,000 meter square identifier, 6 to 10-digit grid coordinate—100 to 1 meter accuracy respectively): `12RWV7040083640`
14. **ISR Source Platform** — Identification of the specific Sensor Platform/Source that collected and reported the information (also known as the bumper/airframe/hull/unit number or name): `2UAUAV`
15. **ISR Source Type(s)** — Identification of the types of sensors used by the ISR Source Platform to collect and report the information. If more than one type of sensor is used, the combination of sensors will be reported within parentheses separated by a space. Single source report: `SIGINT`; multiple source: `(MTI SAR)`
16. **Source Latitude** — Location of source along the parallel of latitude (North or South, Degrees, Minutes): `N,40,42.033`
17. **Source Longitude** — Location of source along the meridian of longitude (East or West, Degrees, Minutes): `E,47,06.946`
18. **Source Altitude** — Height of source Above Ground Level (AGL) in feet: `1053`
19. **Source Military Grid Reference System (MGRS) Location** — Location of source within the MGRS rectangular grid (Grid Zone Designation, 100,000 meter square identifier, 6 to 10-digit grid coordinate—100 to 1 meter accuracy respectively): `12RWV7040083640`
20. **Source Direction of Movement**[b] — Cardinal, ordinal, or azimuth (vector) along which the source is moving (field is blank if source is stationary): `NW` or `270`
21. **Source Speed**[c] — Source velocity in kilometers per hour: `25`
22. **Confidence of Information** — The degree of confidence in the report, expressed as a number in the range 0–100: `80`
23. **Ground-Truth Target ID** — A unique identifier for the ground-truth object (GTO) or aggregate that gave rise to the report: `2_2_8807`. This attribute is available only with simulated data sets and is used only for evaluation—it is ignored in CIFAR processing activities.
24. **Ground-Truth Target Type**[d] — The fully detailed ground-truth BSO or aggregate type according to unit identification or enemy equipment list codes: `2S3`. This attribute is available only with MARS-generated scenarios and is used only for evaluation—it is ignored in CIFAR processing activities.

[a] Provided only in JCATS Version 3 data sets.
[b] Not provided in JCATS Version 1 data sets.
[c] Not provided in JCATS Version 1 data sets.
[d] Provided only in MARS data sets.

Table 1: Data Set Report Syntax

JCATS Data Sets

Data Set	*Region*	*Run Date*	*Reports*
ds05	Yevlakh	5 June, 2006	13,273
ds05a	Yevlakh	5 June, 2006	37,258
ds17	Yevlakh	17 August, 2006	22,122
ds10	Yevlakh	10 October, 2006[a]	54,239

MARS Data Sets

Data Set	*Region*	*Run Date*	*Reports*
dsm19	Yevlakh	19 October, 2006	79,380
dsm13	Baghdad	13 March, 2007	41

Early Data Sets

Data Set	*Region*	*Run Date*	*Reports*
ds18	Yevlakh	18 January, 2005[b]	55,303
ds08	Yevlakh	8 March, 2005[c]	8,895
ds22	Yevlakh	22 April, 2005[d]	8,895
ds11	Yevlakh	11 July, 2005	102,850
ds01	Yevlakh	1 September, 2005	35,185
ds09	Yevlakh	9 November, 2005	61,528
ds30	Yevlakh	30 November, 2005	80,069

[a] Group ground-truth labeling was not provided for this data set.

[b] A preliminary JCATS-generated data set produced by Christian Pizzo. Not all report fields were present and some others contained incorrect values.

[c] A manually annotated data set generated by Chet Brown.

[d] A corrected version of the ds08 data set.

Table 2: Report Data Sets

slowed by ground characteristics. Other entities might react to this by adjusting their own speed and spacing in order to maintain operational guidelines.

There are actually four slightly different report-stream formats in the data sets provided to CIFAR: three for JCATS-generated scenarios and one for MARS-generated scenarios. They differ in the number and content of attributes present in each report item, and CIFAR automatically recognizes which report format is being supplied and processes it accordingly. In addition, the detailed format of some individual items varies from the specification (for example, dates formatted as 5-Jun-06, versus 05Jun06), and CIFAR also adapts to those variants.

Domain knowledge CIFAR obtains problem-domain knowledge from the Knowledge Server in XML format, either live over a socket connection or from XML files if CIFAR is being developed or demonstrated without the Knowledge Server operating.

Note that an individual intelligence report may not specify the target type in terms of a specific, detailed classification. For example, many unattended ground sensors (UGS) might only be able to report the type as either being WHEELED or TRACKED. A novice field observer might report spotting a TANK rather than a T-72B or a IMR-2M.[6] CIFAR needs to understand the relationship and compatibility of intelligence reports with different levels of type-identification specificity, and this is one form of problem-domain knowledge that is obtained from the Knowledge Server. Figure 3 shows the entity-type specificity graph that was used in conjunction with the Yevlakh-region data sets.

[6] An engineering laying vehicle built on a modified T-72 chassis.

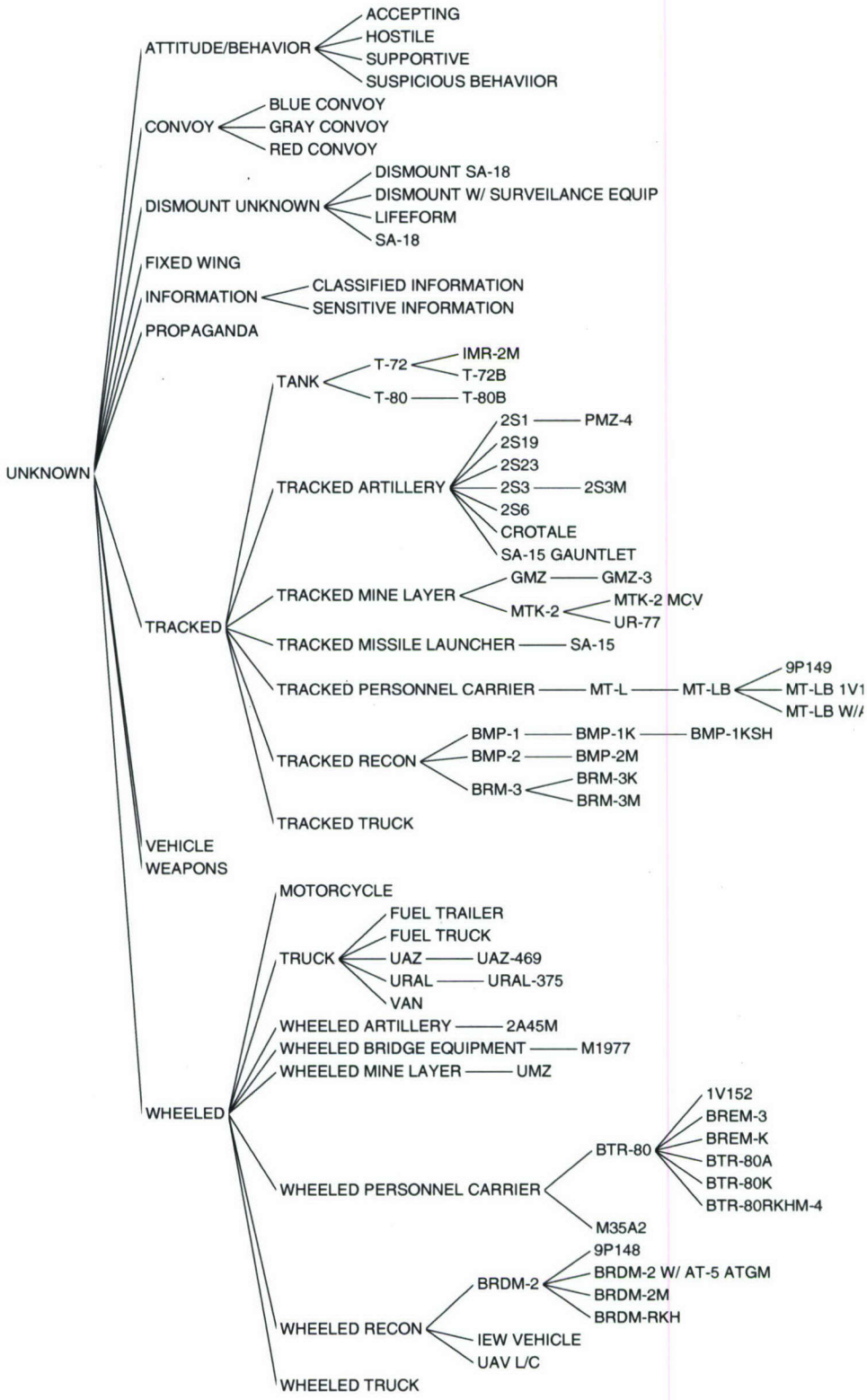


Figure 3: Entity-Type Specificity Hierarchy

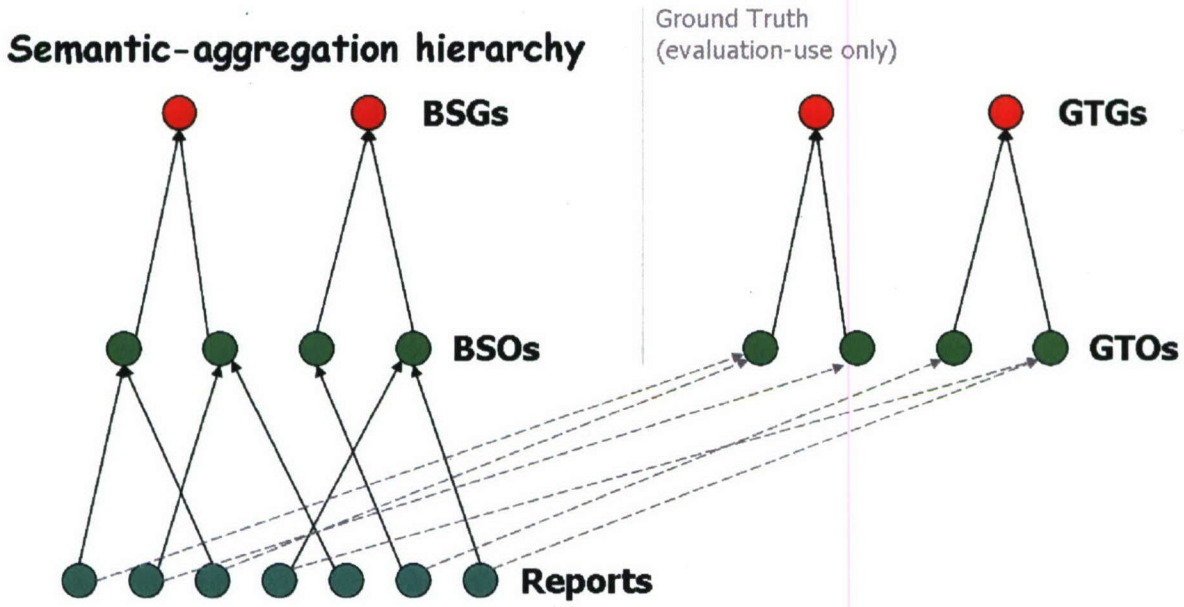


Figure 4: Semantic-Aggregation Hierarchy

PIR/SIR CIFAR also obtains PIR/SIR from the Knowledge Server in XML format. Again this information can either be obtained live over a socket connection with the Knowledge Server or from XML files if CIFAR is being developed or demonstrated without the Knowledge Server operating.

3.2 CIFAR Processing

At a simple level of detail, CIFAR processing involves aggregating individual reports into hypothesized BSOs and then aggregating those BSOs into BSGs (groups of BSOs that are operating together). This multi-level semantic aggregation is is illustrated on the left side of Figure 4. In addition to, and concurrent with, this semantic (and primarily spatial) aggregation is following the movement of these BSO and BSG entities over time. Such temporal aggregation (or "semantic tracking") is a major contributor to assessing the behavior and possible intent of the enemy.

For evaluation purposes, GTO-labeled reports can also be aggregated into GTOs, which can be compared with the BSOs created by CIFAR (shown on the right side of the figure). Note that GTOs do not represent the complete knowledge of ground-truth battlespace objects and their simulated movements, as that information is not fully reconstructable from GTO-labeled reports. Using complete ground-truth information would not be a fair evaluation strategy anyway, as we would not expect CIFAR to intuit BSOs that were never observed in any intelligence report. GTOs represent the best that CIFAR could possibly do given the reports it receives.

Finally, GTOs can be aggregated into GTGs (ground-truth groups) using the provided ground-truth force structure hierarchy for GTOs. GTGs can only be loosely compared with the BSGs, however, as the doctrinal, hierarchical force-structure decomposition knowledge is not directly related to the spatial deployment of BSOs on the battlefield.

A number of factors make CIFAR's processing difficult. As mentioned above, observed entities are reported at varying levels of classification detail. Thus, CIFAR must consider all the possible classification candidates that are compatible with a report (for example, the many possibilities for `WHEELED` or `TRACKED` labeled UGS-generated reports) as well as considering the potential that a report

Figure 5: Are They the Same BSO?

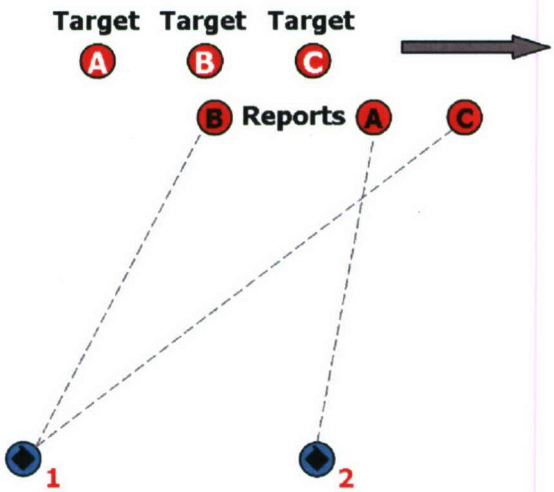


Figure 6: Temporal Skew

has misclassified the target.

BSOs are not continuously observed. Minutes may pass between the last report of a BSO and the next. The longer a BSO is unobserved, the more difficult it is to assert that two reports are of the same BSO (Figure 5). A classic real-world example of this situation is the "tunnel" scenario: "A red car is seen entering one end of a tunnel, and a minute later a red car is seen exiting the other end of a tunnel. Is it the same car?" How long should we watch both ends of the tunnel for other red cars before we are confident that the two are indeed the same red car? (And, are there other entrances that we don't know about? Do people live (and park) in the tunnel? ...)

Not only are there significant time gaps between BSO observations, but the observations are also skewed in time. Figure 6 shows a convoy of three BSOs (`C`, `B`, and `A`) moving from left to right. Sensor 1 observes `B` and sometime later observes `C`. Later still Sensor 2 observes `A`. The observed BSO ordering and their spacing cannot be obtained directly from the three reports without compensating for the sensed-time differences among them.

It can also be very difficult to determine how many BSOs have actually been observed by a set of reports. Consider the two examples shown Figure 7. Figure 7(a) shows reports generated by two sensors that observed the same target. Figure 7(b) shows the same two reports generated by sensors that each observed a different target. Given sufficient positional uncertainty in the reports, it is impossible to distinguish these two situations without additional constraining knowledge or additional observations.

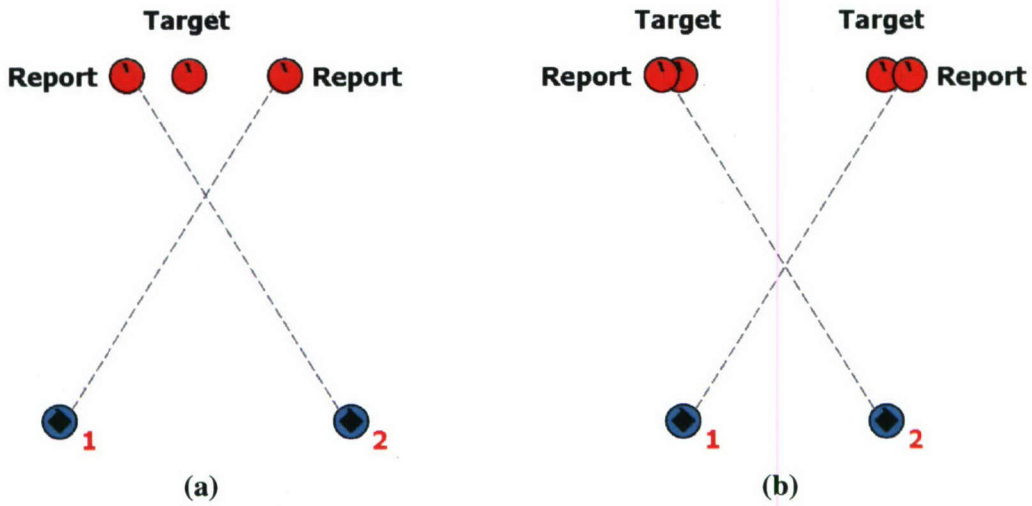


Figure 7: How Many?

Obviously we do not want CIFA alerting the S2 staff that there are 20 tanks moving down the road when there are actually only two. Getting the counts right is a crucial aspect of CIFAR processing.

4 Blackboard-based Temporal and Spatial Aggregation

Aggregating individual reports in both space and time can greatly increase confidence in the count, identity, and behavior of observed entities. Specifically we want CIFAR to assist the S2-team in the identification, composition, and tracking of enemy force-structure components as they move about the battlefield. Temporal aggregation involves associating the positions and movement of individual BSOs and spatially proximate groups of BSOs over a number of observations received over time. Spatial aggregation involves identifying groups of BSOs that are operating together. Temporal and spatial aggregation can greatly clarify behavioral activities that appear uncorrelated and without purpose from the perspective of individual observations at any point in time.

Template-based aggregation In conventional, force-on-force combat settings, enemy BSOs tend to be deployed and operate in fairly structured doctrinal patterns. We felt that applying knowledge of these force-structure behavioral patterns to the intelligence report observations would allow CIFAR to assess the state of the battlefield more quickly and with higher confidence.

Consider the following simple example of using this kind of spatial-deployment knowledge. Figure 8 shows the spatial template (SA01) for a pattern of objects that, when observed together, form an aggregate group of objects (an SAO group). The template represents how individual SAO objects are expected to be related spatially to one another. All SAO instances are expected to consist of the objects shown in the figure as solid circles (objects of type A, B, and two C's). SAO instances may also contain some or all of the optional objects shown as dotted circles (two D's, another B, and an E). All the component objects that do exist are expected to be in approximately the spatial configuration shown relative to the orientation of the template. The layout of Figure 8 represents the pattern objects as if they were viewed from above, with the pattern oriented left (front) to right (rear). Not depicted in the figure is functional knowledge representing the confidence that an SAO has been detected. This confidence is based on how many of the expected objects have been observed, the optional objects that are also present, and the spatial variance of the objects from the template.

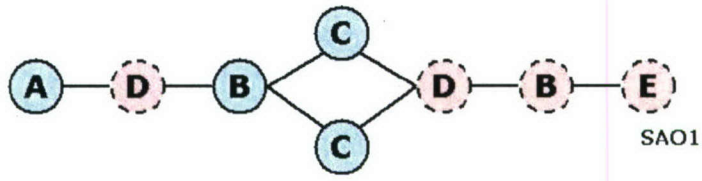


Figure 8: A Simple Spatial-Aggregation Template

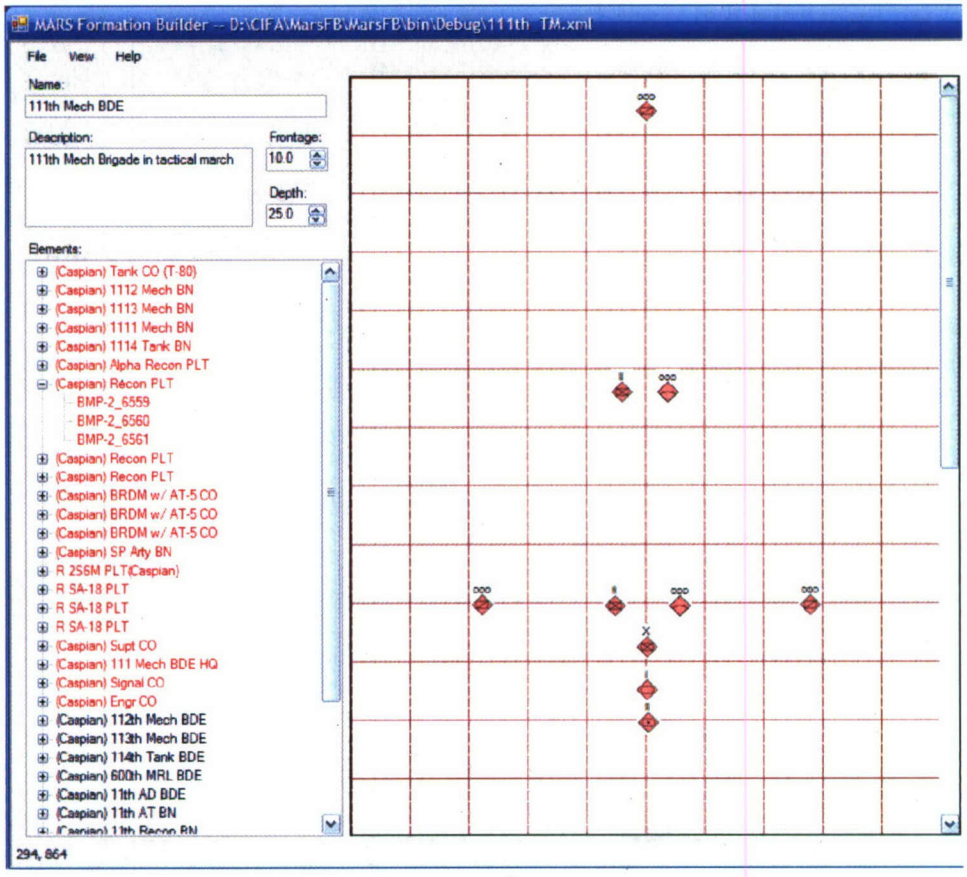


Figure 9: The Leading Portion of 11th Mechanized Brigade as Specified in FormationBuilder (Courtesy BBTech Corporation)

Of course, this simple example is only suggestive of the force-structure template representation and matching process. Actual force-structure templates (Figure 9) include parameters that control the allowed extent variance of subgroups within the template, the distance variance between subgroups, and the angular variance among the subgroups. This recursive "blob-and-spring" (B&S) representation provides a highly expressive representation for spatial relationships and constraints on their adaptation. The straight-line B&S template can be aligned, bent, compressed, or extended based on environmental conditions, and this modified template instance used as the baseline for CIFAR confidence matching.

Uniformed aggregation In contrast to the template approach, uniformed aggregation involves identifying spatial and temporal patterns in the report stream without the use of force-structure pattern knowledge. Instead, uninformed aggregation applies object tracking and clustering methods to identify collective BSO behavior. We believed that uninformed aggregation would be used as the initial step in the process of applying force-structure templates to high-confidence clusters formed by uninformed aggregation methods. Given our belief in this uninformed-to-template-based aggregation strategy, coupled with an initial lack of force-structure template knowledge consistent with the scenario data sets that were available to us, we began focusing the bulk of our aggregation efforts on uninformed aggregation.

4.1 Initial Aggregation Strategies

We started our uninformed aggregation work by evaluating the effectiveness of various standard multidimensional clustering techniques for uninformed aggregation in CIFAR. Clustering is the process of grouping data into partitions ("clusters") on the basis of similarity in their features. Clustering techniques are used in many fields and have been well studied. Unlike the majority of clustering applications, however, we do not have a large static data set to understand. Instead, we want to be able to identify appropriate spatial-temporal clusters using a relatively small amount of intelligence reports (such as those received over a 5 or 10 minute period), and then use new, incoming reports to track those clusters, identify when completely new clusters have been observed, and decide when old clusters should be split up, merged, or deleted.

One commonly used clustering technique is *hierarchical clustering*. Hierarchical clustering methods proceed either by iteratively merging small clusters into larger ones (agglomerative methods, by far the most common) or by splitting large clusters (divisive methods). The result of this process is a hierarchy of clusters, where the hierarchy shows how the clusters are related to each other. A specific partitioning of the objects can then be obtained by cutting the hierarchy at the desired level and taking the newly created leaf clusters as the data partitions. Agglomerative methods use criteria for merging small clusters into larger ones, and often these criteria concern the pairwise merging of clusters (thus producing binary trees). Divisive methods use criteria for subdividing a cluster, and again, a binary subdividing criteria results in a binary tree. A key issue for totally automated application of hierarchical clustering is deciding where to cut the hierarchy so that the resulting partitions are useful. A secondary issue for CIFAR use is determining splitting (or joining) criteria that lead to hierarchies that reflect spatial-temporal report aggregations. Given these issues, we believed that it would be very difficult to obtain reasonable fully-automated report clusters using hierarchical methods.

K-means and related derivative partitioning methods[7] are relatively simple and therefore quite fast when applied to large data sets. The k-means algorithm proceeds as follows:

- choose the number of clusters, k
- randomly select k random points as cluster centers
- assign each object to the nearest cluster center
- recompute the new cluster centers
- repeat the two previous steps until some convergence criterion is met (usually that the assignment hasn't changed)

A well-known disadvantage of k-means is that the resulting clusters depend on the initial random assignments. It minimizes intra-cluster variance, but does not ensure that the result has a global minimum of variance. The main disadvantage of using k-means for CIFAR is that the number of clusters, k, must be specified. Although k-means can be used in an iterative procedure that varies k and

[7]CAST (Cluster Affinity Search Technique) is the graph-theoretic cousin of k-means. It also has the requirement that the number of clusters (cliques) is known in advance.

then uses validity measures to determine the "quality" of the resulting clusters (such as the intra-cluster variance), an approximate estimate of k is still required.

Self-organising maps (SOMs) use a simple type of neural network that can organise high-dimensional data into a low-dimensional (usually 2d) "map" of clusters. Each neuron in the network corresponds to a cluster, and the neural network is utilized to adjust the meta-structure to better represent the clusters. SOMs have the advantage of rapidly constructing clusters that conform to a meta-structure. However, the main drawback of using SOMs for clustering is that this meta-structure, including the number of clusters must be known prior to clustering. Again, this requirement is not well suited to use in CIFAR.

DBSCAN A more promising clustering algorithm for CIFAR was DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [5]. DBSCAN was designed to discover clusters of arbitrary shape and number. Instead of requiring the number of desired clusters, DBSCAN is controlled by two parameters: Eps, the maximum radius of a neighborhood, and $MinPts$, the minimum number of points required in an Eps-neighborhood. DBSCAN works as follows:

- arbitrarily select an object p
- determine all objects that are *density reachable* from p given Eps and $MinPts$
- if p is a *core object*, a cluster is formed
- if p is a *border object*, no objects are density-reachable from p, so select the another object from the database as p
- continue until all objects have been processed

An object is a *core object* if there exists at least $MinPts$ other objects within a radius of Eps from it. An object is a *border object* if it is on the border of a cluster. An object p is *directly density reachable* from another object q if it is within a distance Eps of q and there are at least $MinPts$ other objects within a distance Eps of q. An object is *density reachable* from another object if there is a transitive chain of objects that are pairwise *directly density reachable* with one another.

We began using DBSCAN on the `ds18`, `ds08`, and `ds22` data sets, using only target location (x and y) and sensed time (t) report attributes. We decided to ignore target altitude as it would only be meaningful if certain BSOs were on a ridge and other BSOs were at the base of the ridge. Clustering in only x, y, and t also made sense, as velocity and direction values were not reliable in the preliminary data sets that were provided to us.

Small, spatially well-separated clusters were relatively easy to identify using DBSCAN, but after extensive trials we were unable to find Eps and $MinPts$ settings that worked well on multiple data sets or even on different time periods of the same data set. A known problem with the DBSCAN algorithm is that the use of fixed global values for Eps and $MinPts$ can result in the formation of one giant cluster. Reducing these values can easily transition this one-cluster situation into a situation with far too many clusters. Recursive DBSCAN (RDBC) variants, in which Eps and $MinPts$ can be reduced during processing have been developed, and we experimented with an RDBC strategy as well. However, we remained unable to determine an automated strategy that resulted in consistently useful clusters. As a sanity check in case we were having difficulty due to multiple reports of the same BSOs being received from different sensors, we even tried applying DBSCAN on a per-sensor-platform basis. This sensor-platform-based partitioning showed slight improvement in parameter stability, but introduced a secondary requirement of merging the resulting clusters.

Synthetic snapshots We began to suspect that the irregularity of BSO sightings and temporal skew were making three-dimensional clustering difficult. We could not simply drop the time dimension and cluster in x,y space as there were few reports sensed at any single point in time. In order to move to

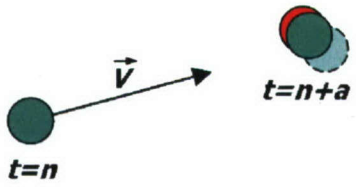


Figure 10: Predictive Tracking

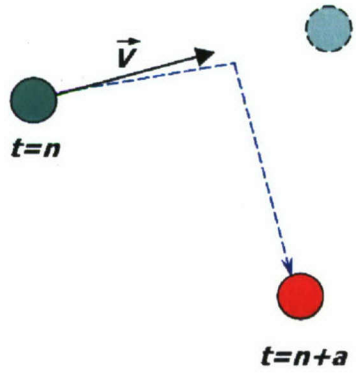


Figure 11: Failed Prediction

two-dimensional clustering we would have to generate a "synthetic snapshot" of the reports at an instance of time by estimating the position of reports that were sensed within a delta-window of the desired time. Fortunately data sets with reasonable target speed and direction attributes had now become available to us, so it was possible to perform constant-velocity extrapolation to time-align reports (Figure 10). The constant-velocity extrapolated position is shown as the dotted, light-blue circle in the figure. The red circle indicates the actual (unknown) position of the BSO at time $n + a$. (The darker blue circle at time $n + a$ will be discussed later.)

These extrapolated reports were represented on the blackboard as extrapolated-report objects. Since the attributes of individual intelligence reports are not fully accurate, variations are expected in the target location, speed, and direction values of a report. Even if the errors in target location, speed, and direction are relatively small, their effect on the extrapolated-report's location becomes increasingly magnified as the amount of temporal extrapolation grows. Further estimated location errors arise if the BSO changes speed or direction during extrapolation interval (Figure 11).

This extrapolation error makes the choice of a synthetic-snapshot frequency all the more important. A long interval between snapshots would create large errors in extrapolated position. Short intervals would both increase processing time due to the larger number of snapshots and reduce the likelihood that a BSO would be sensed at all during a particular snapshot. We experimented with various snapshot frequencies, and 300 seconds (or 5 minutes) seemed to provide a nice balance between cost, BSO presence, and accuracy.

Use of synthetic snapshots allowed us to explore two-dimensional DBSCAN clustering with significantly improved results. However, even with improved cluster identification, the problem remained of how we would track (unify) clusters from one snapshot to the next. We also needed to address the issue of determining the count and target-type identity of individual BSOs from the extrapolated-reports associated with each cluster (the problem illustrated in Figure 7, page 10). A BSO that was reported by three

different sensors at some point during the snapshot interval would result in three extrapolated-reports that are (hopefully) close to one another in the snapshot. We certainly do not want CIFAR to count them as separate BSOs!

4.2 Sequential-Interval (SI) Algorithm

We realized that an important reason that the synthetic snapshot approach improved clustering so dramatically was that we were now implicitly using additional report information in creating the extrapolated snapshots. The speed and direction of a reported target, in addition to its position, help to relate it to other reports. We had been throwing this important information away in our attempts at three-dimensional (x, y, t) clustering. Given this new awareness, we believed that we could make even better use of predictive extrapolation to help merge multiple reports into single BSO-sighting objects based upon their expected location in the synthesized snapshot.

Conceptually this approach proceeds as follows:

- When a new report is received, extrapolate its position at the next snapshot time using its target location, speed, and direction attributes.
- Look to see if a "compatible" BSO-sighting is present within an *extrapolation-distance-delta* distance of the extrapolated report position.
- If a compatible BSO-sighting exists, add the report to the list of supporting reports for that BSO-sighting; otherwise create a new BSO-sighting object at the extrapolated position and add the report as its initial supporting report.

The *delta-extrapolation-distance* value compensates for the likely extrapolated BSO-position error of reports stemming from the same BSO. For snapshot frequencies of 5 minutes, a value of 500 meters worked well.

A "compatible" BSO-sighting is one whose target type is equal to that of the report or to a less-specific parent in the entity-type specificity hierarchy (Figure 3, page 7). If the new report is more specific than the target type of the BSO-sighting, the BSO-sighting type is set to the more specific value.

The 500 meter *delta-extrapolation-distance* value allows reports of nearby compatible BSOs to be merged into a single BSO-sighting. To distinguish between the cases illustrated in Figure 7 (page 10), additional constraint knowledge is needed. One such constraint comes from the characteristics of the reporting sensor platform. Sensors do not report seeing the same BSO continuously. Each particular sensor type has a distribution of how frequently it issues reports for the same object observed within its sensing region. CIFAR can make use of this in making its decision of whether a report should be added to an existing BSO-sighting. If the BSO-sighting contains another report from the same sensor platform that was sensed closer in time than the typical re-sensing frequency of that sensor, then there is strong evidence that the new report is either an error or a second (compatible) target that is close by. In this case, CIFAR will consider the report as incompatible with the existing BSO-sighting and create a new BSO-sighting for the report.[8]

We did not have sensor characteristics for the simulated sensor-platforms used in creating the report data sets. However, we did have GTO report labeling and this allowed us to determine the characteristics of the simulated sensors used in generating these data sets. It turned out that all of the sensors used in the JCATS simulations had the same re-sensing characteristics. The same was true for MARS, but its sensors had a more frequent re-sensing distribution. Figure 12 shows the two re-sensing distributions. For the JCATS sensors, 13 seconds or greater was observed to be the re-sensing frequency 90% of the time (circled in the figure). For MARS sensors, 5 seconds was the comparable value. So, we used 13 seconds and 5 seconds, respectively, as the minimum re-sensing constraint value.

[8]The reports associated with the existing BSO-sighting can then be redistributed among the two BSO-sightings.

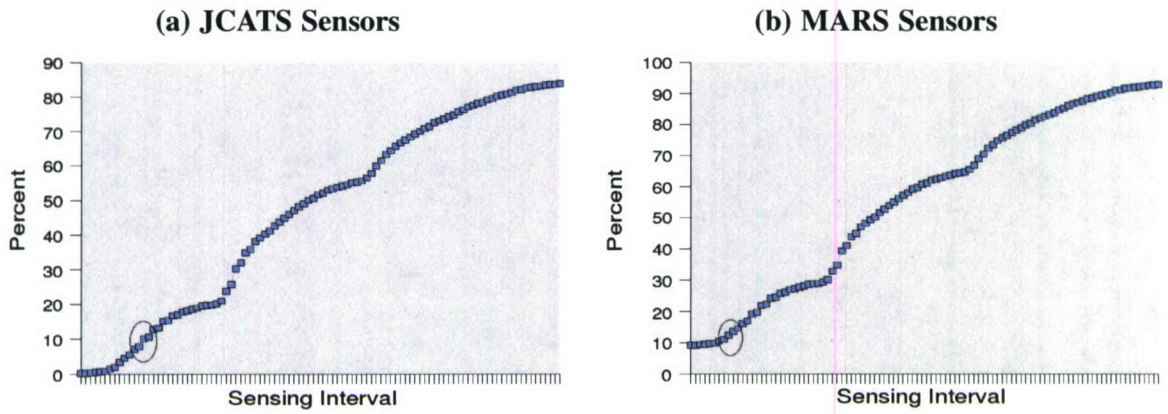


Figure 12: Re-Sensing Frequency

Once the world clock reached the end of the snapshot period, and all BSO-sightings had been generated for the snapshot, the position, speed, and direction values for each BSO-sighting is determined by performing a weighted average of the values in the supporting reports.[9] Then BSG-sightings were generated by clustering the BSO-sightings in the snapshot and their position, speed. Finally, speed and direction values for the BSG-sightings are determined from the supporting BSO-sightings similarly to how they are determined for BSO-sightings from individual reports that support them.

The next step is to update CIFAR's BSO and BSG assessments using the snapshot's BSO-sightings and BSG-sightings. This is accomplished using a similar extrapolation-based tracking algorithm to extrapolate the BSOs and BSGs positions from the last snapshot to their expected positions in the new snapshot. Once again, we face the issue of errors being magnified when extrapolating over a significant time interval. Our experiments, however, indicate that errors in individual reports tend to cancel one another out when determining their values, which reduces the extrapolation error this time around.

BSO and BSG generation Since the tracking algorithm for both BSOs and BSGs are exactly the same, only the BSO generation algorithm will be described. At the initial snapshot, a BSO is generated for each BSO-sighting. The location, speed, and direction of the BSO is that of the BSO-sighting. Once the BSO-sightings and BSG-sightings in a new snapshot have been processed, each BSO from the previous snapshot is extrapolated to its expected location the current snapshot. A search is then conducted over all the compatible BSO-sightings in the new snapshot to locate the candidate BSO-sightings that are in the vicinity of each extrapolated BSO. A minimum distance global search is performed to obtain the best BSO-sighting to BSO pairings. Then, BSO-sightings which have not been paired with a BSO have new BSO objects created for them. BSOs that could not be matched with a BSO-sighting are not updated for the new snapshot. The location, speed, and direction of all other BSO sightings are then updated using values from the BSO-sighting, and the target type of the BSO is made more specific if the BSO-sighting specifies a more specific identification.

This same procedure is used for BSGs, where compatibility is related to the similarity of BSO composition between the BSG-sighting and the BSG. We also observed that CIFAR created BSG-sightings that were very close to each other but which were not being merged into a single BSG-sighting due to the termination condition of our clustering algorithm. These BSG-sightings were so close to one another that BSGs were often being merged and re-split in successive snapshots. In order to better

[9]The weighted average includes the values of all members, with the most recent (least extrapolated) member receiving the highest weight.

identify and track BSGs and minimize this effect, we added a secondary BSG-sighting merge procedure that is run after the BSGs are extrapolated to the new snapshot and paired with BSG-sightings but before new BSGs are generated. The secondary merging was done in two steps:

1. We merged unpaired BSG-sightings which are within 5000 meters of their closest compatible BSG. The reasoning behind this is, if there is a compatible BSG within its vicinity, this BSG-sighting was not attached because another BSG-sighting had been paired with the BSG. Since the unpaired BSG-sighting is also sufficiently close to the BSG, the two BSG-sightings should be viewed as one and merged into this BSG.
2. We then combine any remaining compatible BSG-sightings which are not within 5000 meters of an extrapolated BSG, but are close to other unattached BSG-sightings. This allows us to generate larger and fewer new BSGs for the new snapshot.

4.3 Reverse Sequential-Interval (RSI) Algorithm

Although our SI algorithm was doing a good job of uninformed aggregation and tracking, we observed that the initial positions of newly extrapolated BSO-sightings were being determined by the earliest reports in the snapshot interval—the reports that were extrapolated the furthest ahead in time. Although using the initial reports received in the snapshot period allowed the creation of a snapshot's BSO-sightings earlier, we would gain seeding accuracy if we did not create BSO-sightings until the end of the snapshot and then worked backward from the most recent reports to the earlier ones in the snapshot interval. We found that this approach did moderately increase the accuracy of our BSO-sighting creation over the SI approach.

Since the remainder of our SI algorithm was unaffected by this change to report ordering for BSO-sightings, RSI proceeds identically to SI once the BSO-sightings have been developed for the snapshot.

4.4 Interval-Based BSO-Sighting-Extrapolation (IBSE) Algorithm

The IBSE algorithm is the dual of the RSI algorithm. Instead of operating on each report (in inverse time order) to form the new snapshot's BSO-sightings, we extrapolate the prior snapshot's BSO-sightings to the time of each report as it is received and then use that extrapolation as the basis for associating reports with BSO-sightings. The idea is to take advantage of the potentially more accurate position, speed, and direction values associated with the collectively generated BSO-sighting when doing the extrapolation, as well as shortening the extent of each extrapolation. Consider once again the extrapolation setting shown in Figure 10 (page 14). The extrapolated BSO-sighting location is shown as the dotted, light blue circle. The newly received report is shown as the solid red circle. We could simply update the BSO-sighting that has been matched to the report to the report's location. Or, we could assume that there is some reasonableness in the predicted location versus the potential for an errorful report and weight the two positions, as shown by the solid, dark-blue BSO-sighting circle. This parametrized "predictive inertia" can be beneficial in smoothing out noisy reports at the cost of slower response to slight, but actual BSO movements.

Reports that cannot be matched with an extrapolated BSO-sighting are retained until all reports in the snapshot period have been received and processed.[10] Then, our RSI algorithm is applied to generate BSO-sightings for those reports. IBSE reduces the effect of speed and direction inaccuracies in individual reports, since those reports whose locations are consistent with the extrapolated BSO-sightings can be matched without relying on their reported velocities.

[10]Or at least until the majority of reports have been received. If some reports have long delays from their sensed time to the time they are received by CIFAR. Significantly delayed reports can still be used to slightly update completed processing, but if a large percentage of reports are being received with delay, adding a lag time to CIFAR reasoning may be a better approach than generating premature (and potentially incorrect) BSO and BSG assessments.

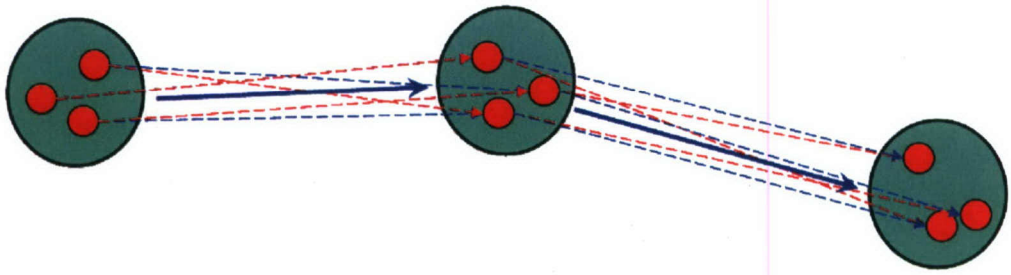


Figure 13: BSG Aggregates Help Tracking

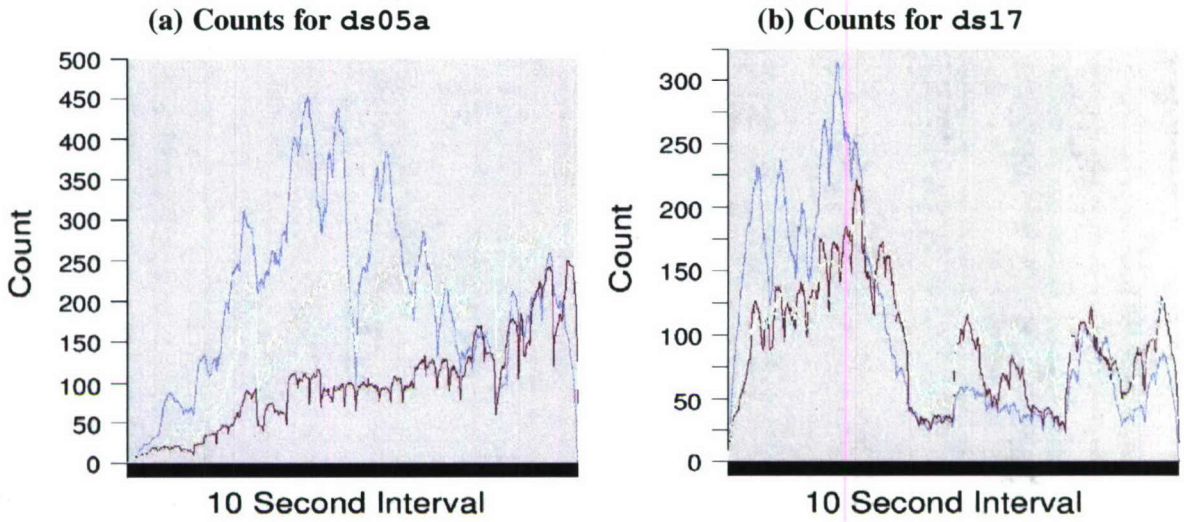
Once again, the changes in our IBSE algorithm dealt with the generation and tracking of BSO-sightings, so most of the remainder of the algorithm is the same as RSI and SI. IBSE does eliminate the need to perform most of the BSO-tracking process, since the majority of BSO-sightings in the new snapshot have been iteratively extrapolated by incoming reports from the past snapshot's BSO-sightings. As a result, IBSE provides an added benefit of lower computational cost than RSI and SI.

4.5 Non-Interval-Based BSO-Extrapolation (NIBSE) Algorithm

The NIBSE algorithm is a natural extension of IBSE. Instead of creating and then iteratively extrapolating new BSO-sightings forward in each snapshot period as incoming reports are processed, we could directly update hypothesized location, speed, and direction of BSO objects based on the reports. This would allow us to avoid creating the intermediate BSO-sightings, except for the fact that we are using them for the BSG-sighting creation and tracking portion of the RSI algorithm. (We will deal with this issue in a moment.) Another advantage of operating directly on BSOs is that we can also eliminate the use of snapshot intervals. We simply update the appropriate BSO whenever a new report is associated with it. If a new report comes in which does not fit with any existing BSO, we can draw one of two conclusions: either a brand new BSO was observed by the sensors or the BSO has moved in an unpredicted fashion as indicated in Figure 11 (either because the BSO did move that way in the world or because the report was in error). In this situation, we create a new BSO based on the maverick report, allowing confirmation by future reports to decide if a new BSO exists in the world or if the new BSO was simply the result of a bad report. The NIBSE BSO-extrapolation algorithm does not attempt to relate new BSOs that result from an unexpected turn by the BSO.[11] This issue is better handled at the BSG level, as all the BSOs in a BSG are likely to change direction in concert (Figure 13). In particular, this figure illustrates that accurate BSG-level assessment and tracking can be performed without necessarily getting the BSO-level tracking correct. There may not be sufficient constraining information available to distinguish the BSO movements shown with the dotted red lines from the alternative ones shown with the dotted blue lines. Nevertheless, the BSG tracking shown by the solid blue arrows remains accurate.

Without BSO-sightings, however, we can no longer use the IBSE BSG-sighting creation, clustering, and BSG tracking techniques. So, NIBSE parallels the direct BSO-extrapolation process with BSGs (eliminating the creation of BSG-sighting objects). Newly created or updated BSOs take the role of new reports in the BSG-extrapolation process. The BSG associated with existing BSOs is updated when the position of one of its BSOs change. If the movement of one or more BSOs becomes inconsistent with the others in the BSG, the BSG is subdivided. If the movement of two proximate BSGs become highly correlated, the BSGs are merged. Newly created BSOs have their own BSG created for them, which will be merged with another BSG if its movement is correlated with it.

[11] The old BSO will soon disappear, as CIFAR will be unable to extend it forward in time.



RSI counts are shown in blue, IBSE in red, and NIBSE in yellow. Actual counts are shown in light blue.

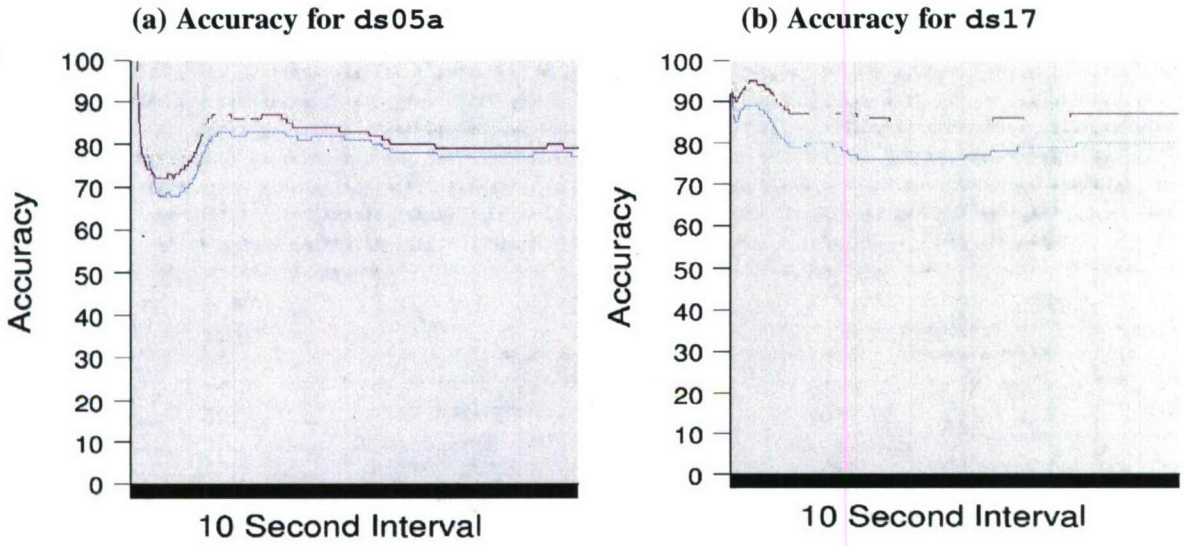
Figure 14: BSO Count Performance

4.6 Summary of Algorithm Results and Conclusions

In this section, we present performance results of CIFAR's use of the algorithms. Because the difference between the SI and RSI algorithms is relatively minor in both accuracy and performance, we will use RSI as representative of both algorithms.

BSO counts The first evaluation looked at how well each algorithm performed in terms of assessing BSO counts throughout each data set scenario. Accurate BSO counts is an important factor for S2 intelligence staff, as the difference between 2 tanks and 20 results in a significantly different assessment of the battlefield. Thus, we wanted to measure how well CIFAR performed in terms of assessing the BSOs at any given point in time. It is important to keep in mind that the "actual BSO count" corresponds only to the BSOs that were actually reported, since we cannot expect any algorithm to identify BSOs that were never reported. The actual BSO count is computed over a sliding 5-minute window called a *retention period*. Retention windows will be discussed later (on page 24), but intuitively they relate to how long CIFAR should consider a BSO as being known following the last time it was reported being observed. Thus the actual BSO count is the number of uniquely GTO-labeled reports sensed during the retention period. The assessed count for each algorithm is the number of BSOs that were created/updated by CIFAR during that same retention period. Figures 14(a) and 14(b) provide a comparison between the RSI, IBSE, and NIBSE algorithm counts and the BSO counts measured over 5-minute retention-period windows slid forward in time every 10 seconds throughout the course of the scenarios.

These results show that the NIBSE algorithm performs the best in both data sets. NIBSE also performed significantly better with MARS data sets, as it is better at handling individual BSO behavior variations than the other algorithms. RSI and IBSE both over-counted at times and both exhibited a high fluctuation in their counts. NIBSE mirrors the progression of actual BSO counts very well. There are a couple of especially good points during the scenarios, where the fluctuation in actual BSO count is well matched by NIBSE. However, there are several points where even NIBSE has difficulty. There are also



RSI accuracy is shown in blue, IBSE in red, and NIBSE in yellow.

Figure 15: BSO Accuracy Performance

occasions where, for a short period of time, the NIBSE count drops briefly even though the actual counts are increasing. It appears that this happens when the GTOs are operating too close to one another, and CIFAR merges these reports into single BSO.

BSO accuracy The second evaluation looks at the target-type accuracy of the three algorithms (Figure 15). The actual accuracy value is computed by counting the most target-type-specific report for each GTO from its first observation through the end of the retention period as the "target-type identity" of that GTO. As before, we cannot expect our algorithms to intuit the specific target-type of a BSO that has only been reported at a generic level. Again, NIBSE outshines RSI and IBSE. It is able to maintain an identification accuracy between 85–90% in both data sets. Note that another important conclusion to be drawn here is that NIBSE performs a successful tracking algorithm. Since the target-type attributes vary in specificity over time, high accuracy in identifying the correct types is possible only by effective long-term unification and tracking of the varying reported target-type specifics.

In summary, we are very pleased (and even pleasantly surprised) with the uninformed performance of NIBSE. These evaluations used only "constant velocity" predictions and yet performed reasonably well on data sets where BSO movements deviated from constant speed and direction. NIBSE is well suited to using more knowledgeable predictions, such as knowledge of travel routes and terrain as well as probable avenues of approach. We expect that the use of more informed predictions would enable more restrictive setting of NIBSE consistency and merging parameters.

5 CIFA/UI: The CIFA Graphical User Interface

To be effective, CIFA must help analysts and decision makers work efficiently and productively within their perceptual and cognitive limits. Experienced, human analysts and decision makers are skilled at quickly assessing complex situations—once they are recognized—focusing on key aspects in making decisions in the face of often incomplete and uncertain information. On the other hand, human decision

makers have great difficulty keeping track of lots of information, remembering details that do not seem immediately relevant, considering alternatives beyond initial reactions, observing and monitoring long-term activities and trends, and multitasking. Automated tools can help in addressing these limitations, as they can incorporate large volumes of data, possibilities, and outcomes in performing assessment and decision-making activities. The goal with CIFA is to use automated capabilities to magnify and augment the inherent capabilities of human operators through delegation and collaboration.

User-interface layout and presentation techniques are an important aspect of achieving effective human-system collaboration. An interactive, map-based presentation of the latest (and recent) assessed positions and movement of enemy forces is particularly useful in conveying CIFA's understanding of the current battlefield situation. This geospatial presentation is provided by the CIFA/UI client (depicted as the yellow "Graphical Situation Representation" box in the CIFA architecture shown in Figure 1 on page 2).).

An initial design for the CIFA/UI client was developed for FalconView,™ a a popular Microsoft-Windows-based mapping system that displays various types of maps and geographically referenced overlays.[12] FalconView was developed by researchers at the Georgia Tech Research Institute and is available free of charge to all components of the U.S. Department of Defense. We were particularly drawn to FalconView's support for a large number of overlay types that can be displayed over any map background. This capability was well suited to our interactive-map presentation strategy, as we wanted to be able to display changing object and threat representations on terrain maps, much as plastic transparency sheets can be used manually in conjunction with physical maps.

After reviewing FalconView's capabilities, we detailed how our interactive UI approach would operate. The CIFAR engine would continuously create, modify, and delete objects on FalconView's dynamic overlays. User-generated events, such as mouse clicks, region marking, and keyboard inputs, would be transmitted from FalconView back to CIFAR to change the display, drill down to details of individual objects, obtain summaries of selected regions of interest, and provide user guidance to CIFAR's reasoning process. FalconView provides a number of APIs (application program interfaces) for third-party extensions, and we chose to use the ILayer and ICallback facilities, both part of FalconView's AutomationInterface. The ILayer interface allows separate-process applications or same-process (shared memory) COM objects to add vector based items to the FalconView map. It also allow the application to control overlay order and to open and close overlays. ICallback is a COM interface that allows FalconView to deliver user-generated events back to third-party programs. When the user interacts with a graphical item that was created by a third-party program using the ILayer interface, FalconView will call the appropriate associated method via the ICallback interface.

Although FalconView's APIs were a good match for our CIFA/UI interactivity goals, Distributed Common Ground Systems (DCGS) conformance was an overarching requirement, and we migrated the FalconView design to C/JMTK (the Commercial Joint Mapping Toolkit).[13] C/JMTK is a standardized geospatial toolkit of software components for the management, analysis, and visualization of map and map-related information. Within DCGS-A, C/JMTK provides Common Operating Environment (COE) mapping, charting, geodesy, and imagery (MCG&I) functionality.

Research programmer Ken Watts was brought into the project to support CIFA/UI development with C/JMTK. He began by importing Yevlakh-region map data provided by RDECOM's Christian Pizzo into C/JMTK. To reduce the huge amount of disk storage required for the map data, Ken manually generated map overlays of the specific areas that were used in the simulated scenarios. This allowed the CIFA/UI client to be easily installed on laptops or desktops that had limited free disk space available.

The predominate use of C/JMTK for display is presentation and interaction with static mapping

[12] http://www.FalconView.org/

[13] http://www.cjmtk.com/

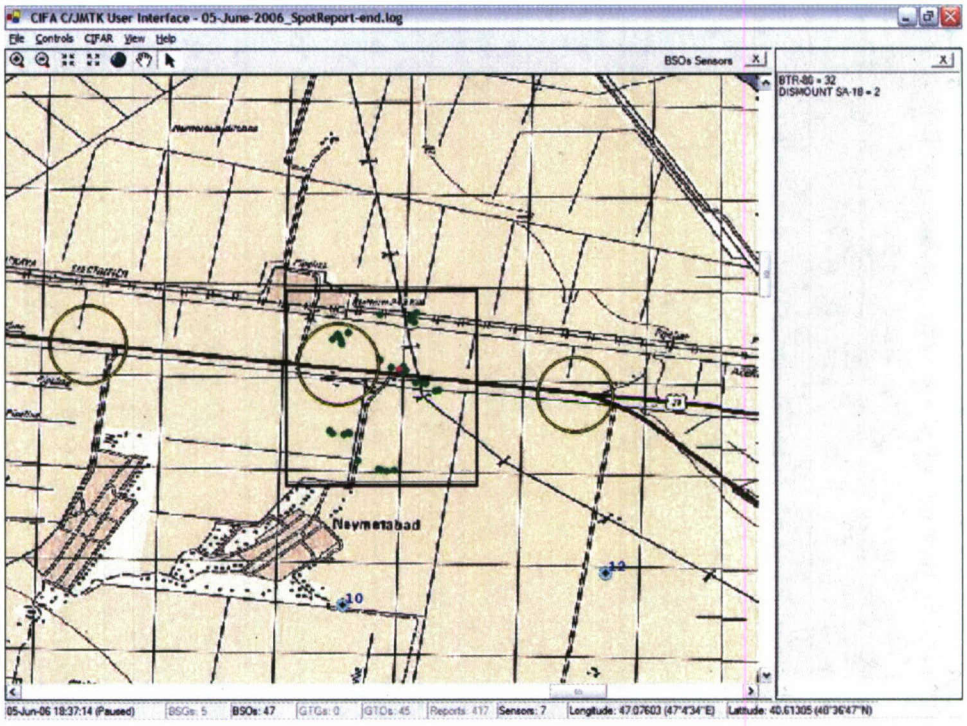


Figure 16: CIFA/UI Showing Aggregated BSO View

information (or dynamic information that has been preprocessed in advance of user presentation). Our envisioned up-to-the-second interactive UI was atypical of C/JMTK use. Ken explored (with Tom Quinn, a member of the C/JMTK support staff) the most effective C/JMTK-based approach for our needs. There were several options suggested at the time. The first option involved using the ESRI tracking server, which was thought to be able to handle the movement of several thousand objects in real-time. The only problem was that the tracking server has not been government funded for use with C/JMTK–although the tracking server "might" be supported at some time in the future. Another suggested option was to use C/JMTK's tracking layer, which was designed to present predetermined, file-based, dynamic tracking information. However, there were some significant problems associated with the tracking layer approach including performance issues when displaying more then a few hundred objects.

Given the problems with these suggested approaches, we elected to perform direct overlay-layer manipulations from a .NET (C#) application, using some of the same C/JMTK interfaces that we assumed were being used by the ESRI tracking server. Obtaining technical information on these interfaces was difficult, as our desired use of C/JMTK was unusual (and not covered in the public documentation) and our requests for details fell outside of the routine C/JMTK support questions. Ken investigated a number of ESRI code examples and "works in progress" and, along with considerable trial and error, developed an acceptable dynamic-overlay-style interface to C/JMTK.[14]

The result is the CIFA/UI display shown in Figure 16. This display is showing the current BSO assessments in a small region of the battlefield that were made by CIFAR operating on the `ds05` JCATS-generated scenario. Our goal was to keep the CIFA/UI interface consistent with C/JMTK style operations. The menu bar at the top left contains drop-down items that can be used to configure the

[14]We were not alone in dealing with these issues. Bailey and Odom highlight the some of the differences and challenges that they face in their work toward integrating FalconView and C/JMTK [6].

CIFA/UI client and to customize the display presentation.

The large map pane in the figure is showing individual BSO positions as small solid-color dots. Also shown as small circles with diamond centers (toward the bottom of the map) are the positions of reporting sensor platforms. The larger unfilled circles depict SIR regions of interest that are actively being monitored by CIFAR.

At the top of the map pane is a tool bar that allows the user to zoom in or out in various ways, to drag the map (via the hand icon), or to select individual objects or regions (using the arrow icon). At the right side of the tool bar, the types of CIFAR objects being displayed are shown (currently BSOs and sensor platforms).

What is displayed on the map pane can be customized by the user. In addition to selecting the general CIFAR types of objects being displayed, the user can elect to display only certain target types or only BSOs that have been reported by specific sensor types or specific sensor platforms. Object-display colors can be assigned to highlight specific target types, and so on. These customizations are performed easily using the `View` drop-down menus.

To the left of the map pane is the information-display (or "console") pane, where detailed textual information is displayed. In the figure, the user has selected a region on the map (shown as black rectangle) by clicking and dragging the mouse, requesting that a summary of the BSOs in the region be displayed on the console pane. In this case, there are 32 `BTR-80s` and 2 `DISMOUNT SA-18s` hypothesized to be in the selected region.

At the bottom of the CIFA/UI display is a status bar that displays the current world-clock time, the current number of different object types, and the latitude and longitude of the mouse cursor. The counts of object types not being displayed are shown in light gray, as a reminder that they have not been selected for display on a map pane.

Note that the current time display indicates that CIFA is paused. In an operational setting, CIFA must operate in real time. For research and demonstration purposes, however, it is very useful to allow CIFA to run much faster than real time, consuming reports at a multiple of actual clock time, and to pause the world to discuss a CIFA feature or to investigate a behavior. The CIFA/UI provides this type of time-base control. Using the drop-down customization menus, the time-base of CIFA can be adjusted and the processing can be paused and resumed by pressing the `P` key to toggle CIFA's "paused" state. Rather than wait hours for a battle scenario to advance to an important condition, CIFA can be allowed to run at high speed until the condition approaches and then slowed to real time in order to observe the condition. This style of demonstration and research is possible because the CIFAR reasoning engine is able to perform its computational activities in a fraction of real time on the data sets that were available in this effort.

It is also useful to review events that led up to the current situation. CIFA/UI also supports such history review. The display clock can be set backward to a specific or relative ("30 minutes ago") time, with the review progressing at whatever display rate is desired. When a CIFA/UI client is performing such a review, CIFA itself is paused, if in simulated-reports mode, or continues to progress in real time, if in operational mode. The CIFA/UI client can be advanced forward to "the present time" at any point.

Figure 17 shows the same paused scenario as Figure 16, but with hypothesized BSGs displayed rather than BSOs. In this case, there is one BSG in the map area, drawn as a green unfilled ellipse that suggests the BSGs approximate latitude and longitude (lat/lon) extent. The BSG object was selected by the mouse, and its description is printed in the console pane. Notice that there are discrepancies between the equipment compositions shown for the BSG and the region summary counts shown in the BSO-view figure. Explaining this difference requires a bit more detail of CIFAR processing and the CIFA/UI display philosophy.

Recall that BSOs are rarely constantly observed. CIFAR maintains BSO objects at their last position

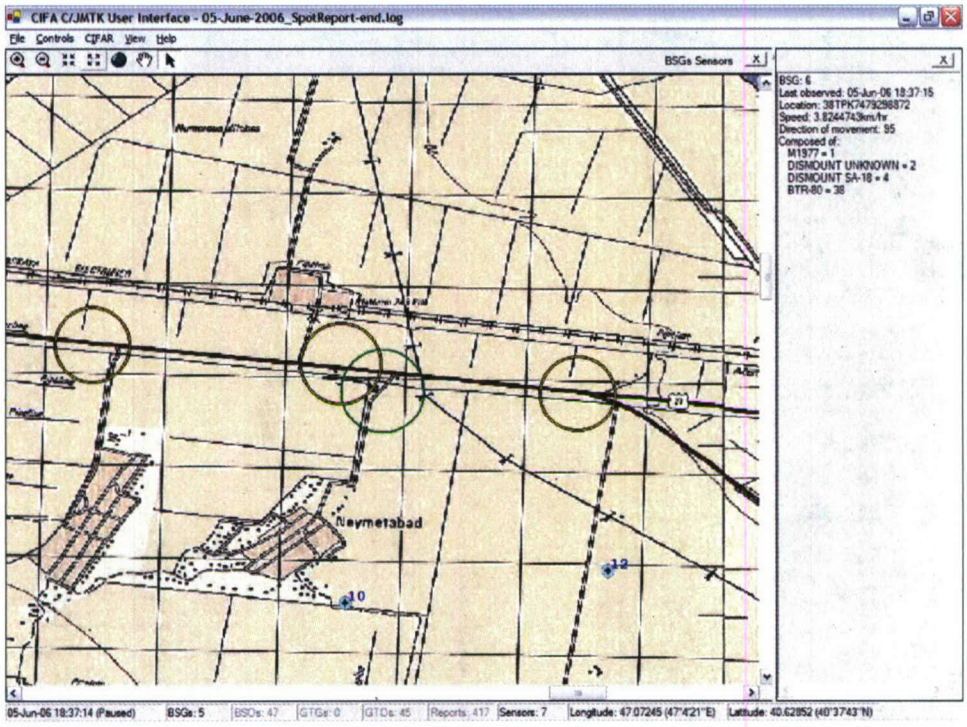


Figure 17: BSG View

supported by actual reports, and this positioning is what is displayed by CIFA/UI clients. Thus, an individual BSO will appear to "jump" from an old position to a new one, as new reports support a new location for the BSO. We could have elected to use expectations to smoothly animate the assumed location of each BSO on the map by projecting it forward in time from its the last reported position. This could be done within the CIFAR engine itself or by the CIFA/UI clients, if the expectations are provided to them. We elected to be conservative in CIFA, by maintaining and displaying information based solely on received reports and not on extrapolation of expectations.

What happens if no reports are received for a BSO for a period of time? If we kept displaying the BSO at its last known location, it remain where it was last shown indefinitely. This would eventually mislead the user into thinking that the BSO might still be at that location, even though no recent observations have been received that support that assumption. To address this, we added a *retention-period* capability to CIFA. Objects that have not been updated by at least one report within the retention period are not displayed on the CIFA/UI, even though the information about them is still retained in CIFAR. The retention period is specified using the drop-down configuration menus in CIFA/UI, with values ranging from 5–15 minutes being typical for the scenarios used in this effort.

So, where is the `M1977` BSO that is listed as a component of the BSG but is not shown in the BSO figure? The answer is that it has not been reported within the retention period, and so its actual location since it was last reported is unknown. The hypothesized BSG still considers it possible that the `M1977`, as well as 2 `DISMOUNT UNKNOWN`s, 2 `DISMOUNT SA-18`s, and 6 `STR-80`s that had been moving previously with the other BSOs in the BSG are still moving with them even though those extra BSOs have not been sighted within the retention period. If they continue to be unobserved, the confidence that they remained part of the BSG will soon drop sufficiently that they will be eliminated from the hypothesized components of the BSG. The BSO view does not take the BSG aggregation into account

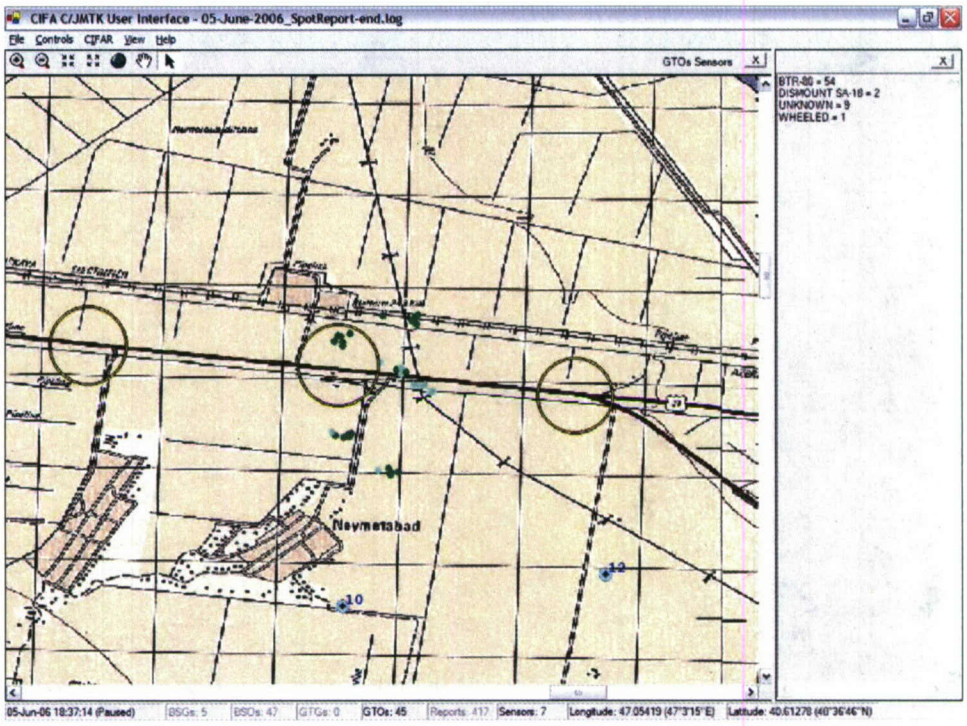


Figure 18: GTO View

when displaying hypothesized BSOs–only individual BSOs that have been updated during the retention period are shown. Note that it remains possible to obtain the last position of the `M1977` in the CIFA/UI client, either by drilling into the details of the BSG's components (which include those extra BSOs) or by increasing the retention period of the CIFA/UI client which will add less recently updated objects to the current display.

The CIFA/UI client can also be used to display ground-truth labeled (GTO) objects, as shown in Figure 17. This display shows the last position of each BSO that was reported within the retention period. Of course, GTOs are not available in operational settings, but this display is useful for understanding how well CIFAR is doing in assessing the simulated scenario. The inventory being shown in the console display is not the summary of GTOs (there are a total of 45 GTOs being shown), but an indication of separate BSOs that were supported by reports of the same GTO. The higher the totals, the more times a BSO was supported by reports labeled from multiple GTOs.

Finally CIFA/UI can also display all the reports received during the retention period, as shown in Figure 17. This display indicates what an analyst could have to consider without the assistance of CIFA and its CIFAR engine. The inventory shown in the console shows the totals of the target-type labels of the displayed reports. Note that without the temporal and spatial aggregation provided by CIFAR, estimating the actual BSO target types and counts is quite difficult.

A CIFA/UI capability that we found particularly useful is split-screen mode. Split-screen enables the user to display a secondary view of the map if desired, as shown in Figure 20. Although we considered supporting separate "world times" in each map pane, we decided to maintain "world time" synchronization of both views. The two views can be "spatially locked" in terms of position and zoom level, so that adjusting the map view in either map pane affects both views. Unlocked views allow different regions and zoom scales to be displayed on the two map panes. The types of objects that are

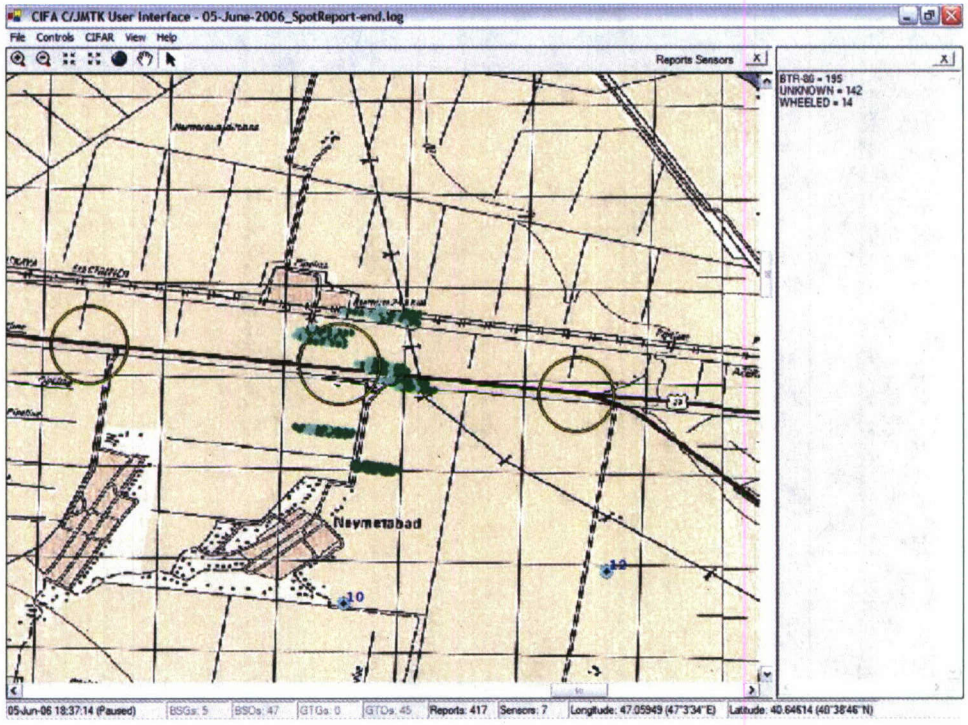


Figure 19: Raw Reports

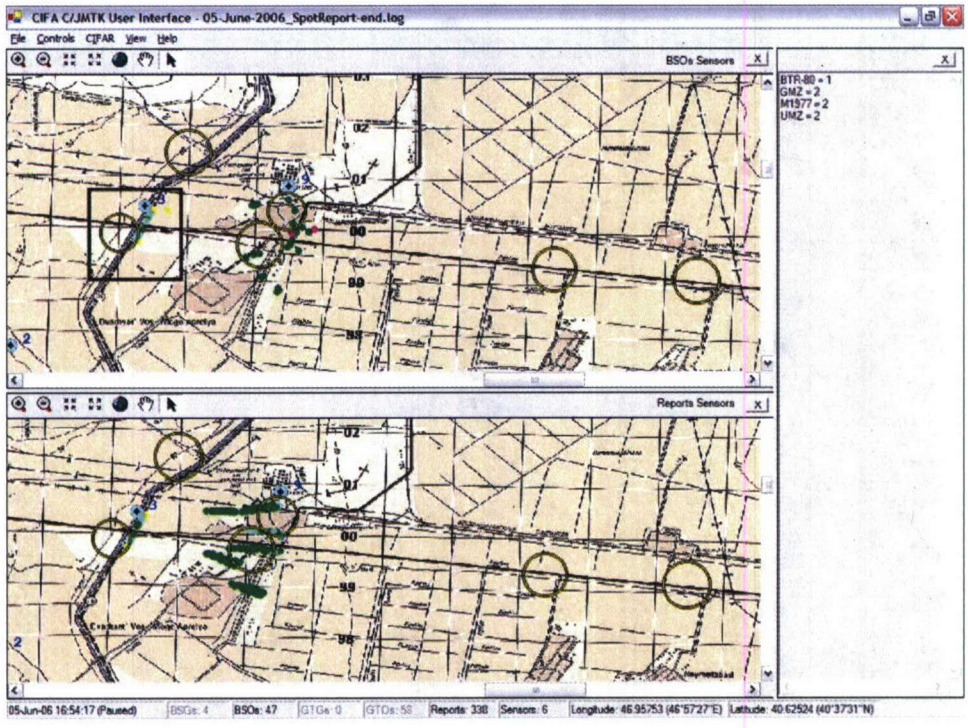


Figure 20: CIFA/UI Split-Screen View 1

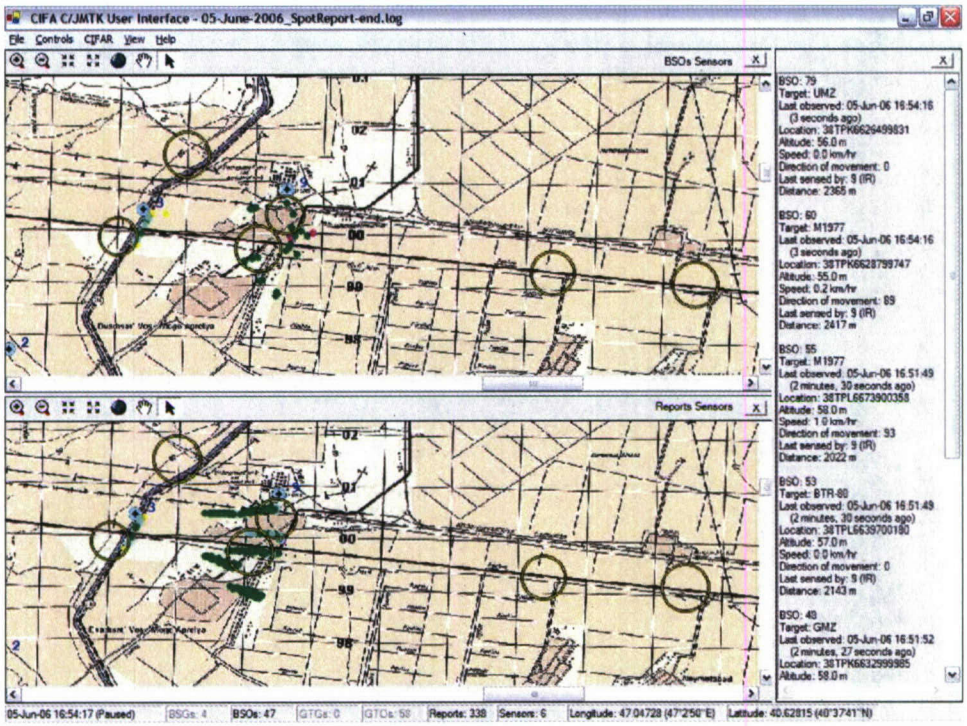


Figure 21: CIFA/UI Split-Screen View 2

displayed in each pane can be set individually. For example, Figure 20 shows BSOs and sensors in the top map pane, and raw reports and sensors in the bottom map pane. Similarly, we have used split-screen mode to display BSOs and GTOs in separate map panes (or BSGs and GTGs). This allows quick visual inspection of how well CIFAR is operating relative to ground-truth-labeled reports.

All of the map-pane operations that are available in single-map mode are also available in either map pane in split-screen mode. Each map pane has its own toolbar for selecting how the mouse behaves in that map pane. It can be handy, for example, to have the drag (hand) behavior enabled in one map pane and the mouse selection behavior enabled in the other. The console panel in Figure 20 shows the assessed equipment counts for the region selected in the upper map pane. The console in Figure 21 shows the BSO details for this same selected region.

6 Event Recognition and Notification

CIFAR automatically notifies PIRManager when a hypothesized BSO (or BSG) or set of BSOs (and/or BSGs) matches the criteria for an SIR. Figure 22 shows the SIR monitoring and notification screen from PIRManager. In this screen shot, there are two SIR that have been matched. When a match is reported, the CIFAR column of the SIR is outlined in red and the text reads "Matched." This situation is seen with SIR ID 1419 in the figure. The SIR is interested in Rebel Atropian forces with `BTR-80`s changing formation in NAI 38. If CIFAR determines that it has observed behavior that matches this SIR, it will notify PIRManager. For example, if an UGS reports that there are one or more `BTR-80`s in NAI 38, this will lead to `BTR-80` BSOs that match the criteria for STR ID 1419. PIRManager highlights the SIR and its attached PIR to inform the intelligence staff that a potentially relevant BSO (or set of BSOs) has been observed. In a similar fashion, SIR ID 1411 was matched at some point, but now the match conditions are no longer being observed. When a match is no longer active, the outline color of the CIFAR column

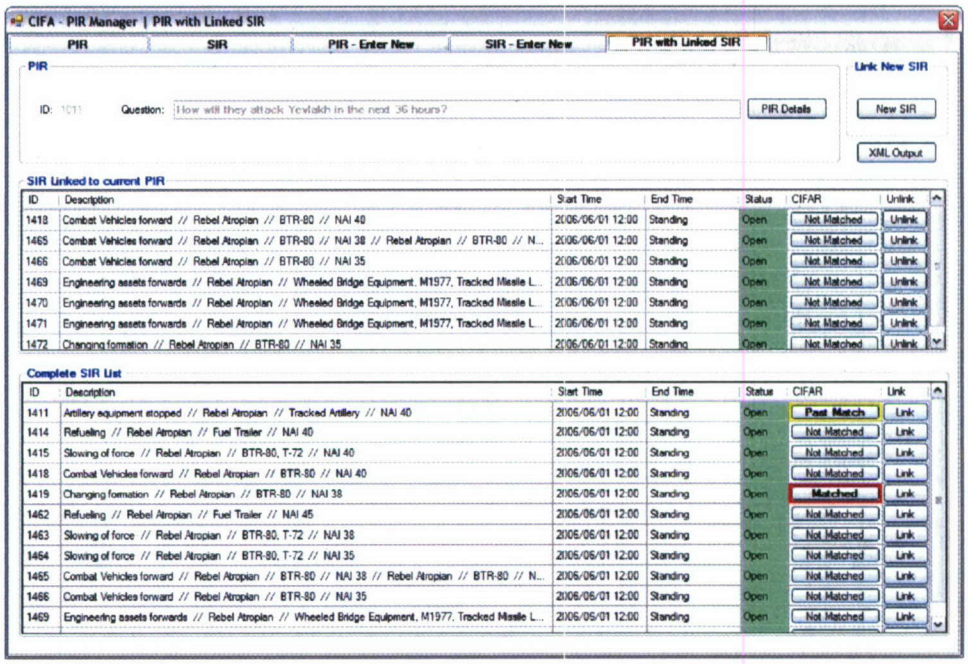


Figure 22: The CIFA PIR/SIR Monitoring User Interface (Courtesy BBTech Corporation)

is changed to yellow and the text changes to "Past Match." In this case, the SIR requests reports of Atropian tracked artillery stopped in NAI 40. The match may no longer be active because there was formerly artillery in NAI 40 but it moved into another NAI. It may also no longer be active because the artillery has not been observed within the current retention period set by the user.

Although it is up to the S2 staff to determine if a significant event has occurred or is still occurring, CIFAR, via PIRManager's PIR/SIR monitor, provides alerts about potentially relevant events that have been observed. The user can retrieve a report summary by clicking on the CIFAR column and can then look at the detailed reports associated with the SIR if desired. By providing automated support for matching intelligence reports with SIR, CIFAR aids the S2 staff in sifting through the potentially overwhelming amount of information that is being collected. The goal is to assist the staff in quickly finding and focusing on high value information.

7 integration of Sensor Data, Reports, and Automated Processing Results

Blackboard applications, such as CIFAR, have historically relied on belief (or "confidence") values in relating estimates of the certainty of input data and knowledge-source (KS) contributions. The extensive use of graphical models, such as Bayesian networks, in modern AI applications [7, 8, 9] has led to criticism of the ad hoc, and often domain dependent, confidence values used in traditional blackboard-system representations [10, 11, 12, 13] and, by implication, of the blackboard applications themselves. These ad hoc confidence values were involved in everything from making control decisions to determining solutions and the system's confidence in them. This criticism generated a burst of interest in developing what have been termed *Bayesian blackboard* systems. The idea is to replace ad hoc representations of the relationships among blackboard objects with incrementally generated graphical models. Recent techniques in constructing belief networks using network fragments [14, 15] and in hierarchical object-oriented Bayesian networks [16, 17] have been suggested as candidate technologies that can be extended to create more "principled" blackboard reasoning.

Preliminary efforts in applying graphical belief networks to blackboard systems have focused on a principled representation of the developing *solution* on the blackboard [18]. The blackboard application's current beliefs are represented on the blackboard as a possibly disconnected graphical network. A first-order extension to belief networks can be used to collapse similar entities into a single node-type and set of arguments that in combination uniquely specifies a node on the blackboard. Time complicates graphical-network representations, and can involve significantly different temporal scales. To address this, multiple temporal representations have been used: a discrete approach where each node is indexed by the time it occurs[15] and a duration-interval approach where nodes have a start and an end time that are themselves represented as nodes in the network. Complex spatial representation and reasoning are also problematic for graphical-network approaches, and *procedural KSs* are often used to perform geometric reasoning. Such reasoning is both difficult and highly inefficient to represent explicitly using a Bayesian-network fragment. These concessions to complexity weaken the foundation of principled reasoning sought by a Bayesian approach and real-world situation assessment representation and inferencing requirements present considerable challenges to the use of Bayesian techniques.

We believed that the sudden emphasis on developing a principled blackboard representation of the developing solution is misplaced and falls far short of addressing the complete set of issues that arise with blackboard systems and other forms of collaborative reasoning. Instead of focusing on blackboard representation, the emphasis should be on making the *integration* of the contributions made by diverse entities well founded. This can only be achieved by understanding and representing how these contributions are generated and how they relate to one another. Blackboard applications work incrementally, with independently developed KSs adding their contributions to those already present on the blackboard. For example, if two KSs use the same data and produce similar results using different computational approaches, how independent are the results? Are they redundant (with no added certainty in the results) or complementary and corroborating (in the sense that each has the potential to make mistakes on certain data values, but these mistakes are fully independent of one another)? In the latter case, contribution integration needs to reflect the additional certainty that is produced by the corroborating contribution. Note that recording the pedigree of the data used in making contributions is insufficient in understanding how they relate. To date, so called "more principled" blackboard-system approaches have simply assumed independence of contributions or used fuzzy averaging and other ad hoc approaches to contribution integration, while professing to have placed blackboard-system processing on a solid, formal foundation.

Unfortunately the overall confidence-integration problem is not solved simply by using network-fragment representations and influence combinations. Consider the two graphical-network fragments shown in Figure 23 (a) and (b). Sensor A and Sensor B are attempting to detect the existence of a vehicle on a bridge and both are affected by the weather conditions. When these two fragments are combined, they will form the network shown in (c), without requiring the use of any influence combination method. However, this implicitly assumes that Sensors A and B are conditionally independent given weather and vehicle. However, such an assumption may not be correct. For example, Sensors A and B may be the same type of sensor and some unknown factor not captured by the nodes weather and vehicle may cause them to both make the same errors, requiring a new influence link between the two sensors, as shown in (d).

Given partial results, each with an associated confidence that the partial result is correct, how should a blackboard system compute the confidence in an integrated result? Figure 24 illustrates a very simple example of this problem represented as a Bayesian network. The node E is a possible event in the world that we are trying to assess. The nodes A_1 and A_2 represent contributions made by two entities. For example, A_1 and A_2 could represent two different sensors that detect whether or not event E occurred. The confidence in A_i is represented by $P(E = t|A_i = t)$. The question is whether or not these two

[15] Making the graphical network a Dynamic Bayesian Network [19].

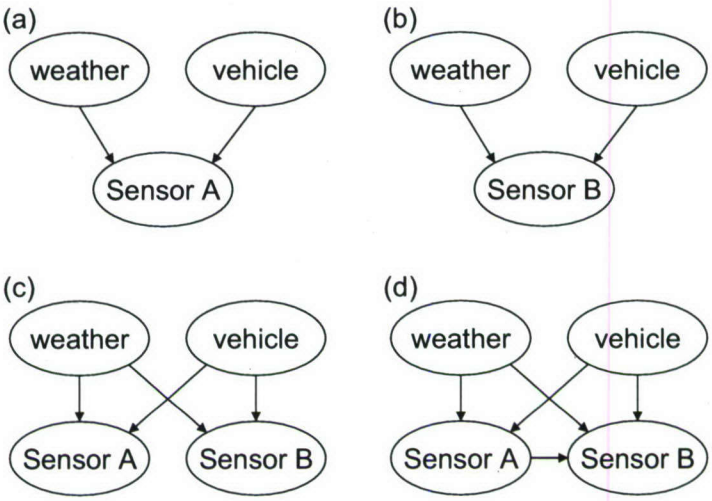


Figure 23: Combining Network Fragments

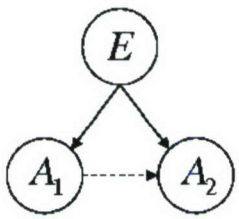


Figure 24: The Confidence-Integration Problem

sensors are conditionally independent or whether they tend to make similar, correlated mistakes. In other words, should there be an influence link between them in the network model (the dotted line in Figure 24), and how strong an influence should that link represent?

This confidence-integration problem shows up in many different places in a system like CIFAR. One example is where there are many different sensors collecting data. Consider a sensor monitoring a bridge. The sensor reports that it has detected the presence of a tracked vehicle with a confidence of 80. In other words, there is an 80% chance that a tracked vehicle actually crossed the bridge.[16] Now suppose that we have another sensor monitoring the same bridge and it also reported sensing a tracked vehicle with a confidence of 80. Now how certain should we be of a tracked-vehicle assessment? The answer depends on how correlated these two sensors are. If they always make the same mistakes at the same time, then our confidence should remain unchanged by the second sensor report. On the other hand, if these two sensors are independent of each other our confidence of a tracked vehicle could be arbitrarily high, depending on the prior probability of there being a tracked vehicle on the bridge (for example, 98.5% for a prior probability of 20%). Even in this simple example, the potential for significant error in the integrated confidence assessment is quite high if we do not take dependencies into account.

The complexity of KS processing and representations at different levels of detail and abstraction in

[16]Throughout this report, we will assume that the "confidence" in a contribution reflects the probability that the contribution is correct.

blackboard-system applications only serves to mask the fundamental issue of determining the confidence of integrated contributions. Even if we know the accuracy of individual contributions, we need to account for the confidence uncertainty inherent in integrating them together and how that confidence uncertainty propagates over long reasoning chains and KS interactions. Note that the integration problem is not solely a blackboard-system issue. Integrating contributions received from others is also an essential activity in collaborative multi-agent systems (MASs) [20, 21] and other collaborating software systems [22, 23] and in real life [24]. How often do jurors intuitively assume contribution independence when multiple eye witnesses testify that "The defendant is the one I saw running from the crime scene."?

Confidence integration and contribution independence As part of this effort, we developed a general model for contribution-based reasoning. We then used this model to perform formal analyses of how the confidence errors that result from incorrectly assuming independence of contributions when it is not present and vice versa propagate in complex reasoning systems. We showed that this confidence error can be significant in applications where the accuracy of input data and processing is not close to perfect. Our formal analysis also highlights where in these systems it is most critical to operate with the proper understanding of dependence implications. The details of this work are discussed in our AAMAS[17] 2007 paper (reproduced in Appendix C), which presents our confidence-integration models and analysis work from a MAS perspective.

One intriguing use of the model that is not discussed in the AAMAS paper, is the potential of using the model for learning the independence of contributions in an open-contributor setting.[18] Intuitively, the approach is as follows. Assume that the accuracy of each of the collaborating entities has been measured as the entity worked alone with test data that has the same characteristics as the overall system will be working with. (In other words, that the test data is drawn from the same distribution as the operational data.) Also assume that the contribution-integration model of the system is correct in terms of its structure and parameters. Then, given these invariants, it is possible to note the correlation of results as they are being contributed by the entities and use this correlation to hone in on the dependence among the contributors. The intriguing aspect of this approach is that it is possible to learn these dependencies on the fly in the running application—*without* ground-truth labeling of the operational data.

How is this possible? The key is that the individual entity accuracies have been measured previously (if separately) using test data with known ground truth. Since the distribution of the test data and the operational data are assumed to be the same—and the integration model is assumed to be correct—only the degree of dependence among the contributors can explain statistically significant additional correlation of contributed results. For example, if two entities exhibit a higher degree of correlation than can be attributed to their individual accuracies within the model, then the additional correlation must be from correlated errors. In other words, the amount of correlated errors among entities is the only free variable that is available to explain the additional observed correlation in entity contributions.

We did not pursue this idea formally in this effort, and the issue of validating the accuracy of the system-specific contribution integration model (in addition to having the individual entity accuracies known) may make such an approach impractical for use in real-world applications. Nevertheless, it remains an interesting direction for further research.

Contribution integration and report confidences In our confidence-value integration model, we assume that the contributions can be integrated trivially. For example, the semantic integration is simple

[17] International Joint Conference on Autonomous Agents and Multiagent Systems.

[18] An "open contributor" or "open system" is one in which individual entities can be added, removed, or improved at any time. In such applications, we do not have the luxury of hand crafting a new interaction and interdepency model whenever the entity population changes.

if we want to integrate three sensor reports of a T-72B tank that has been observed at the same location and time. Only assessing the confidence value in the integrated T-72B BSO hypothesis is difficult. On the other hand, if the contributions are not semantically identical, additional inferencing is required to produce the integrated result and this work also affects its confidence value assessment. Assume that one of the sensor reports identifies the target type as only TANK and another only as TRACKED. How much should we reduce our confidence in a T-72B assessment? Now what if the three reports have slightly different sensed times and are not spot-on in reporting the identical location? Now CIFAR has to decide what different (possible but potentially mutually incompatible) assessments fit the reports as well as the confidence values to assign each of them. These decisions all stem from the confidence values of the original three sensor reports.

This leads us to another CIFAR issue: the confidence assessments of individual reports. Recall from the report structure shown in Table 1 (page 5) that each intelligence report has an associated overall confidence value. For CIFAR to have any hope of performing "principled" reasoning and assigning probabilistically grounded confidence values to its assessments, the confidence assessments in the input reports themselves must be semantically well founded. If a report is received with a "confidence" of 80, what does that mean? Does it mean that the report has an 80% chance of being completely correct? If that is the intent, does it mean that all of the report's attributes will be correct 4 times out of 5? If some of the attributes are incorrect 1 time out of 5, which ones and in what way? The current report format that contains only a single confidence value for the entire report does not allow the following cases to be distinguished:

- "I'm very sure that I saw something big go through the intersection (at x,y), going east (the direction of the road), at close to 50km/hr), but I'm not as sure that it was a T-72B."
- "I just saw a T-72B very near to my location (x,y), but I didn't get it's speed or heading as I was ducking for cover."
- "UGS 8407 reported a faint detection of a TRACKED object that passed over its x,y location."

Representing and using the semantically different certainty characteristics represented in these reports is at the foundation of effective CIFAR reasoning. The different semantics in these examples could be expressed by providing confidence values for the accuracy of each important attribute (or for a set of attributes such as location or velocity) in the reports. Without more detail, it is impossible to know if a report's confidence is low because the target-type assessment is weak, because the location or velocity of the target is uncertain, or if it unclear if anything was observed at all.

8 Lessons Learned

In this section, we reflect on some of the broad lessons learned in performing this research effort.

Representative data sets It is crucial to have "realistic" data sets of one or more scenarios available for this kind of research. A realistic data set should exhibit the character of battlefield scenarios, but it does not need to involve highly accurate sensory data or entity behaviors. What is important is that the scenarios and data sets require CIFAR to perform the same kinds of reasoning (using the same kinds of knowledge and constraints) that would be required in an operational situation. Thus, realistic data sets can be produced that are open (non-classified), with their reports reflecting nominal sensing capabilities and on well-known, open doctrine, enemy behaviors. What is important is that the data sets reflect the level of report loading and uncertainty that will exist in real situations.

For example, data sets generated from simple behavioral ground-truth movements that are augmented (obfuscated) with random missed detections, false identifications, shifts in reported location, and other forms of loss and "noise" are not realistic. Such uncorrelated data can be quickly eliminated by

temporal-spatial observation techniques. On the other hand, data sets that include reports of neutral as well as enemy entities, correlated sensory degradation in a geographic area due to environmental conditions or active obscuring, and so on, make aggregation and assessment decisions much more difficult.

Lack of representative "simulated" ISR data was a major issue in the initial stages of this effort, and data sets that included reports of realistic gray entities and some reported activity descriptions remained out of reach throughout this effort. The need to apply detailed constraint knowledge, such as the re-sensing frequency of sensor platforms, highlights the importance of having realistic data during development and testing of CIFA. Without actual, historical, battlefield intelligence reports (the real thing), a sufficiently detailed and representative modeling and simulation capability is required to drive this form of research. This requirement is even more critical in asymmetric settings, where the reported behavior of enemy combatants in conjunction with that of the civilian population is key.

Representative Knowledge It is also important to incorporate into CIFAR detailed domain knowledge that is consistent with simulated data sets. To the extent that CIFAR is a knowledge-intensive system, that knowledge must encompass and be consistent with the behaviors that occur in the scenarios. If there is a research and demonstration goal of supporting distinct context-specific knowledge based on specific locale and enemy forces, representative data sets can be used to demonstrate CIFAR's ability to incorporate different knowledge bases as appropriate to each setting.

C/JMTK Our experience with C/JMTK in creating the CIFA/UI was not a particularly happy one. We found that using C/JMTK as an underlying toolkit for creating live, interactive displays was cumbersome and resulted in unnecessarily high computational overhead. This may be due, in part, to the lack of publicly documented APIs for using C/JMTK in such a dynamic context. We only needed to use a fraction of C/JMTK's capabilities to produce a responsive CIFA/UI client, but it was in this fraction that C/JMTK came up frustratingly short for us. It will be interesting to see over the next few years how well C/JMTK is able to compete with the capabilities that are being added to such public tools as Google Maps, Microsoft's Visual Earth, and Google Earth. It will be truly unfortunate if the state-of-the-art in military mapping tools does not keep pace with what is available to the general public.

9 Remaining Technical Issues & Recommendations for Future Work

9.1 CIFAR

As mentioned earlier, we are very pleased—if even somewhat surprised—at how well CIFAR is able to identify, aggregate, and track entity activities using primarily uninformed aggregation methods. We had assumed that highly constraining force-structure deployment knowledge, terrain, and route information would be required to make sense out of the high-volume intelligence report stream. With CIFAR's current performance baseline in place, we have the potential to make CIFAR even faster and more accurate if this additional knowledge is provided.

One unanswered question is how CIFAR would perform with a significant percentage of unaffiliated (gray) entity reports and reports of entities of unknown affiliation. In the data sets that were available to us, the intelligence reports were only of enemy BSOs.

We also expected that opportunistic control of CIFAR's reasoning would be required in order to assess the incoming reports effectively in real time. However, the CIFAR engine is currently able to pursue all plausible lines of reasoning at a rate much faster than real time when operating on the supplied data sets.[19] However, the addition of reports of a significant number of gray entities moving

[19]This is even true for CIFAR running on a relatively old and slow laptop.

in proximity with enemy forces could quickly reintroduce the need for opportunistic blackboard-based control strategies.

CIFAR is effective because it makes use of all the knowledge and constraints that are available to it. Currently CIFAR is primarily responsible for offloading from human analysis the identification, counting, and tracking of enemy entities and groups of entities that are operating together on the battlefield. Inferring the intent behind these movements is left to the S2 intelligence staff. Extending CIFAR with additional knowledge and reasoning to assist with intent assessment is a natural direction for further work. Availability of richer intelligence reports (some with more detailed activity descriptions in them) as well as other forms of input (such as communication activity, potential courses of actions and the reasons behind them, and likely enemy objectives) would be required as part of achieving this additional level of decision support.

9.2 CIFA/UI

In spite of the difficulties in using C/JMTK for interactive map-based presentation of dynamic geospatial objects, the resulting CIFA/UI client is very useful at conveying CIFAR's current assessment of BSOs and BSGs on the battlefield. Much of the CIFA/UI development stemmed from the research and development needs of this effort, and we have not collected significant feedback from analysts about what additional capabilities or changes would improve the effectiveness of using CIFA/UI as an operational tool by S2 intelligence staff.

One important enhancement that we have identified is the ability to name and save custom presentation settings (including pane sizes and positions, display filters, color choices, and so on) so that these settings could be retrieved quickly for future sessions or for changing the desired display easily in the current session. This capability would be relatively simple to add to the CIFA/UI, but we did not have the resources to implement it during this effort.

Another enhancement is greater use of military symbology in the presentation. Map-based BSO displays typically do not have sufficient screen real estate to draw military symbols for each BSO unless the user has zoomed into a very small region. It would be good to have CIFA/UI change its object display method from dots to symbols when the zoom level makes this feasible. Similarly when showing more aggregate GTG-level views, displaying force-structure symbols rather than, or in addition to, the current ellipses would be beneficial.

A significant deficiency in the current CIFA/UI is its display performance, especially when CIFA is operating much faster than real time. Much of the computational cost of operating the CIFA/UI client appears to be associated with lat/lon position translation. Internally, CIFAR uses a localized Cartesian coordinate system to make spatial computations of the scale of an MGRS grid square (or less) fast. CIFAR's local coordinates can be translated to MGRS coordinates very quickly, however translating local coordinates or MGRS coordinates to lat/lon values involves complex trigonometric computations. C/JMTK is built to accept lat/lon values, which CIFAR has for report locations (as both MGRS and lat/lon are provided with reports), but not for computed locations (such as hypothesized BSO positions) that are not identical to any report. We implemented lat/lon translations for the CIFAR server, but we felt that it would be better to perform translations between MGRS and lat/lon on the CIFA/UI clients. We discovered a C/JMTK library that allows C/JMTK to operate with MGRS values, but we soon learned that the coordinate conversion is noticeably slow. Apparently the C/JMTK conversion process goes from MGRS to lat/lon to the internal coordinates used within the ArcMAP engine, rather than by a direct MGRS to internal process). To achieve top performance, avoiding lat/long conversion by going from CIFAR internal to MGRS to C/JMTK internal would be highly beneficial and going directly from CIFAR internal to C/JMTK internal would be even better.

9.3 Contribution Integration

Developing a principled and probabilistically sound formalism for integrating contributions made by multiple entities in an open system setting is a challenge that is likely to remain incompletely solved for years to come. If it is not an "AI hard" problem, it is at least a highly difficult one. Formal AI models and techniques have now been developed for principled representation in AI problems, and researchers are beginning to advance these representational models into *causal* models. (Causal models go beyond representing the probabilistic influences that are present in the models to causality influences. An example of this distinction is inferring the causality that smoking causes cancer rather than simply discovering that smoking and cancer are related statistically.) Formal, principled, modeling of contribution integration requires *inferencing* models that represent not only the underlying representation or causal model, but also the process that was used to instantiate that model and its parameters. There is certainly lots to do in this area, and our work in this project is only an initial "baby step."

Although a complete, formal basis for contribution integration is likely to remain unsolved for quite some time, there is no need to wait for significant formal breakthroughs to be achieved. Practical and reliable (if not fully "principled") applications are being developed and deployed using ad hoc and problem-specific representations and techniques (just like what was done in the maligned "classic" blackboard-system applications). What is important, however, is being aware of the assumptions and limitations built into these techniques so that everyone understands when the system is operating in a "nearly principled" manner and when confidence in its inferences should be accepted only cautiously. Our formal analysis research is an example of efforts that can be used to bound the error stemming from the use of "unprincipled"—but effective—techniques.

Acknowledgments

Dr. Gerald M. Powell (U.S. Army RDECOM CERDEC I2WD), Major Chester F. Brown (USAI C&FH), and David Lorenz (EWA Government Systems, Inc.) provided many hours sharing background information and answering questions. Their experience, advice, patience, and enthusiasm for this work were important contributors to the progress we made in this effort. RDECOM's Christian Pizzo, Kevin Prudente, and Jacqueline May provided terrain and JCATS-generated report data sets. Kevin also assisted in digging for undocumented C/JMTK programming details when direct inquiries through official channels went unanswered.

It was also a pleasure working with BBTech Corporation on CIFA. Dr. Susan Lander, Steven Hoffman, Michael Rudenko, and Mathew Bart maintained their adaptability and enthusiasm as we all co-evolved CIFA designs, component representations, and APIs to meet our joint requirements.

On the UMass Amherst side of things, graduate students Paul Kohler and Huzaifa Zafar made contributions to CIFAR's clustering and aggregation algorithms. Former graduate student Raphen Becker agreed to put the final stages of his dissertation research on hold to work on contribution integration. Research programmer Ken Watts was responsible for the C/JMTK extensions to support the CIFA UI. Finally, MAS Laboratory Business Manager and Grant Administrator Michele Roberts kept the diverse administrative and reporting aspects of this collaborative effort under control and running smoothly.

Thanks to you all!

References

[1] Daniel D. Corkill, Kevin Q. Gallagher, and Kelly E. Murray. GBB: A generic blackboard development system. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1008–1014, Philadelphia, Pennsylvania, August 1986. (Also published in *Blackboard Systems*, Robert S. Engelmore and Anthony Morgan, editors, pages 503–518, Addison-Wesley, 1988.).

[2] Daniel D. Corkill. Countdown to success: Dynamic objects, GBB, and RADARSAT-1. *Communications of the ACM*, 40(5):848–858, May 1997.

[3] Sandra E. Keene. *Object-Oriented Programming in Common Lisp*. Addison-Wesley, 1989.

[4] Gregor Kiczales, Jim des Rivieres, and Daniel G. Bobrow. *The Art of the Metaobject Protocol*. MIT Press, 1991.

[5] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (KDD '96), pages 226–231, Portland, Oregon, August 1996.

[6] Chris Bailey and Joel Odom. Integration of FalconView and C/JMTK. In *Proceedings of the Twenty-Sixth Annual ESRI International Users Conference*, San Diego, California, August 2006.

[7] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[8] Judea Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2000.

[9] Spirtes, Glymour, and Scheines. *Causation, Prediction and Search*. MIT Press, 2nd edition, 2001.

[10] Lee D. Erman, Frederick Hayes-Roth, Victor R. Lesser, and D. Raj Reddy. The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty. *Computing Surveys*, 12(2):213–253, June 1980.

[11] Daniel D. Corkill. Blackboard systems. *AI Expert*, 6(9):40–47, September 1991.

[12] Robert S. Engelmore and Anthony Morgan, editors. *Blackboard Systems*. Addison-Wesley, 1988.

[13] V. Jagannathan, Rajendra Dodhiawala, and Lawrence S. Baum, editors. *Blackboard Architectures and Applications*. Academic Press, 1989.

[14] Robert P. Goldman and Eugene Charniak. A language for construction of belief networks. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, 15(3):196–208, 1993.

[15] Kathryn Blackmond Laskey and Suzanne M. Mahoney. Network fragments: Representing knowledge for constructing probabilistic models. In *Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence* (UAI-97), pages 302–313. Morgan Kaufman, 1997.

[16] Daphne Koller and Avi Pfeffer. Object-oriented Bayesian networks. In *Uncertainty in Artificial Intelligence: Proceedings of the Thirteenth Conference* (UAI-97), pages 302–313, San Francisco, California, 1997.

[17] Avi Pfeffer, Daphne Koller, Brian Milch, and Ken T Takusagawa. SPOOK: A system for probabilistic object-oriented knowledge representation. In *Proceedings of the Fourteenth Annual Conference on Uncertainty in AI* (UAI-99), pages 541–550. Morgan Kaufman, 1999.

[18] Charles Sutton, Clayton Morrison, Paul R. Cohen, Joshua Moody, and Jafar Abidi. A Bayesian blackboard for information fusion. In *Proceedings of the Seventh International Conference on Information Fusion* (Fusion 2004), pages 1111–1116, Stockholm, Sweden, June 2004.

[19] Kevin P. Murphy. *Dynamic Bayesian Networks: Representation, Inference, and Learning*. PhD thesis, University of California at Berkeley, Berkeley, California, July 2002.

[20] Jeffrey M. Bradshaw. *Software Agents*. AAAI Press & MIT Press, 1997.

[21] Jacques Ferber. *Multi-Agent Systems: Toward a collective intelligence*. Addison-Wesley, 1998.

[22] Barbara J. Grosz. Collaborative systems. *AI Magazine*, 17(2):67–85, Summer 1996.

[23] Daniel D. Corkill. Collaborating software: Blackboard and multi-agent systems & the future. In *Proceedings of the International Lisp Conference*, New York, New York, October 2003. (Invited presentation.).

[24] James Surowiecki. *The Wisdom of Crowds*. Doubleday, 2004.

A Installing and Running CIFAR

CIFAR and its supporting GBBopen software are provided on a CD as two archives: one containing CIFAR and the other the open-source GBBopen framework.

Before installing the CIFAR and GBBopen software, install a supported Common Lisp implementation.[20] Although GBBopen is provided on the CD, the latest Subversion (SVN) snapshot archive for GBBopen can also be found in the "Downloads" area of the GBBopen web site (`http://GBBopen.org/`). The on-line GBBopen hyperdoc reference and the GBBopen Reference Manual (in PDF format) can be found in the "Documentation" area of the web site. Links to supported Common Lisp implementations can be found on the "Current ports" page of the GBBopen web site. In either case, download or extract the GBBopen files to an installation directory of your choosing and follow the installation instructions in the GBBopen Tutorial that is included in the archive (and is also available on the GBBopen web site.)

Next, create a subdirectory named `gbbopen-modules` in your home directory. (Your "home" directory can be ambiguous on machines running a Microsoft Windows operating system. The result returned from the Common Lisp function `(user-homedir-pathname)` is the directory that your Common Lisp implementation considers to be your "home" directory.)

Next, extract the CIFAR archive into a directory of your choosing.

Then, create a symbolic link in your `gbbopen-modules` directory (the directory that you created as a subdirectory of your "home" directory (above) that points to the directory where you installed CIFAR (the directory where the `commands.lisp` and `modules.lisp` files for CIFAR are found).

Start your Common Lisp and load the GBBopen's `gbbopen-init.lisp` file. (If you have set up GBBopen according to the "Enhancing your Development Environment" exercise in the *GBBopen Tutorial*, this will be loaded for you automatically when you start your Common Lisp.

Then, at the read-eval-print prompt, enter: `(cifar :create-dirs)`. You only need to specify this the first time you install CIFAR. The `:create-dirs` option tells GBBopen to automatically create directories for holding the compiled CIFAR files. After this initial compilation, you can compile and load CIFAR simply by entering the command `:cifar` after the GBBopen's `gbbopen-init.lisp` file has been loaded.

To run CIFAR stand alone on one of the data sets, enter the data set name as a function. For example: `(ds05)`. A special "partial" data set, `(ds05p)`, is provided as a very brief test of CIFAR execution.

To run CIFAR as a server in order to allow processing to be initiated and monitored from a CIFA/UI client, evaluate the function `(start-server)` once CIFAR has been loaded.

[20] CIFAR server services require a multiprocessing Common Lisp to support CIFA/UI and PIRManager clients.

B Installing and Running the CIFA Graphical User Interface

The CIFA Graphical User Interface (CIFA/UI) runs in conjunction with C/JMTK, and is available only for Microsoft Windows XP platforms.[21] We have created a installation DVD that automatically installs the C/JMTK components needed for CIFA/UI, the C/JMTK compatible Yevlakh-region map data, the .NET runtime, and the C/JMTK client. Once installation is complete, simply open the `CIFAUI` executable to launch the CIFA/UI client.

Note that a CIFAR server must be available in order to display anything useful on the CIFA/UI client. The host running the CIFAR server can be specified using the CIFAR drop-down menu. The CIFA/UI client remembers the prior CIFAR-server setting and automatically attempts to contact that server the next time that the CIFA/UI client is launched.

[21] CIFA/UI has not been tested with Windows Vista operating systems.

C Determining Confidence: The AAMAS-07 Paper

This appendix contains a copy of the AAMAS-07 paper: "Determining Confidence When Integrating Contributions from Multiple Agents," by Raphen Becker and Daniel D. Corkill. This paper appeared in the *Proceedings of the Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Honolulu, Hawaii, pages 449–456, May 2007.

Determining Confidence When Integrating Contributions from Multiple Agents

Raphen Becker*
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01002
raphen@gmail.com

Daniel D. Corkill
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01002
corkill@cs.umass.edu

ABSTRACT

Integrating contributions received from other agents is an essential activity in multi-agent systems (MASs). Not only must related contributions be integrated together, but the confidence in each integrated contribution must be determined. In this paper we look specifically at the issue of confidence determination and its effect on developing "principled," highly collaborating MASs. Confidence determination is often masked by ad hoc contribution-integration techniques, viewed as being addressed by agent trust and reputation models, or simply assumed away. We present a domain-independent analysis model that can be used to measure the sensitivity of a collaborative problem-solving system to potentially incorrect confidence-integration assumptions. In analyses performed using our model, we focus on the typical assumption of independence among contributions and the effect that unaccounted-for dependencies have on the expected error in the confidence that the answers produced by the MAS are correct. We then demonstrate how the analysis model can be used to determine confidence bounds on integrated contributions and to identify where efforts to improve contribution-dependency estimates lead to the greatest improvement in solution-confidence accuracy.

General Terms

Measurement, performance, experimentation, theory

Keywords

Contribution integration, confidence, analysis models

This work is supported by the "Fusion Based Knowledge for the Future Force" ATO program and the "Advanced REsearch Solutions - Fused Intelligence with Speed and Trust" program at the U.S. Army RDECOM CERDEC Intelligence and Information Warfare Directorate, Fort Monmouth, NJ, under contract W15P7T-05-C-P621. The views contained in this paper are the authors'.

*Now at Google, Inc., Mountain View, California

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

1. INTRODUCTION

Integrating contributions received from others is an essential activity in multi-agent systems (MASs) [2, 6], blackboard [3, 5] and other collaborating software systems [7, 4] (where problem solving is performed by multiple knowledge sources), and in real life [12]. Not only must related contributions received from other agents be integrated together, but the confidence in each integrated contribution must be determined. The issue of confidence determination and its effect on developing "principled," highly collaborating MASs is the focus of this paper.

Consider a simple "thought experiment" MAS application. There are three "observer" agents, each able to observe a coin toss and report whether it saw a "heads" or a "tails." A fourth "integrating" agent receives the observer reports and combines them to produce an overall answer. Each observer agent has undergone extensive certification of its coin-flip acuity and reports the flip correctly 80% of the time. We wish to improve the accuracy of the four-agent MAS over that of a single observer agent by having the integrating agent combine the reports received from the three observers.

To clarify the issue of confidence determination, we make the following assumptions in this paper:

- trust [8] is not an issue—agents always do their best to provide accurate contributions
- agent contributions are not always correct—despite its best efforts, an agent's contributions may be wrong
- received contributions can be integrated trivially—contributions to be integrated have identical semantics
- the probability of an agent's contributions being accurate is known (0.8 in our thought experiment)
- the "confidence" in a contribution reflects the probability that the contribution is correct

Suppose the integrating agent receives a *heads* report from each observer agent. What confidence should the integrating agent assign to the integrated *heads* result? Common approaches used when combining contributions are to: 1) assume independence among the contributions, 2) use ad hoc heuristics to approximate dependencies, 3) avoid the issue by using only a single ("best") contribution, or 4) not determine a confidence for the integrated contribution at all.

Our integrating agent could assume that the contributions are independent, apply Bayes rule, and assign the integrated result a confidence of .985 (reflecting the collective corroboration of the three contributions). But what if the contributions are not independent? In this thought experiment, let's implement each of our observation agents as a simple

function that is given the actual (ground truth) value of the toss and reports that value except for 20% of the time when it reports the opposite side. The decision of when to report an incorrect observation is determined by using a pseudo-random number generator. Our observer accuracy is 80% (and we've saved a lot of coding). If each agent uses a different seed for its random-number generator, the times when the agents' reports are incorrect is independent and the integrating agent's confidence assignments (based on assumed independence) are realistic. However, if all observer agents use the same seed, their pseudo-random numbers will be identical and they will be mistaken at the same times. If our integrating agent accounts for this contribution dependency, the confidence in the integrated result should remain at 0.8 no matter how many observer reports are integrated. (The reports are fully redundant.)

Of course, real MAS agents are not about random-number generators and ground-truth cheats, and the interactions among agents are often significantly more involved than sending complete results to an integrating agent. A real agent-based coin-flip detector might consist of a number of camera agents (each with its own camera), low-level image processing agents (with different processing approaches and algorithms), feature-detector agents (*Eye*, *Nose*, *Hair*, *Building*, *Head*, etc.), side-assessor agents (again, potentially with different knowledge/strategies), and so on. Yet this complexity only serves to mask the fundamental issue of determining the confidence of integrated contributions. Even if we know the accuracy of individual contributions, we need to account for the confidence uncertainty inherent in integrating them together and how that confidence uncertainty propagates over long reasoning chains and agent interactions. A "principled" integration agent in our thought experiment should represent the confidence in the result as the interval [0.8..0.985], with any additional knowledge of the dependency characteristics of observer contributions applied to reducing this confidence interval.

The remainder of the paper is structured as follows. First, we introduce a domain-independent analysis model that can be used to measure the sensitivity of complex, collaborative problem-solving systems to potentially incorrect confidence-integration assumptions. We then demonstrate how the analysis model can be used to determine confidence bounds on integrated contributions and to identify where efforts to improve contribution-dependency estimates lead to the greatest improvement in solution-confidence accuracy. We conclude with future directions to explore in contribution integration.

2. THE ANALYSIS MODEL

We have developed a Bayesian network model [9, 10] that facilitates analysis of confidence integration in an arbitrarily complex collaborative problem-solving system. While the analysis model is Bayesian, the system being modeled can have any representation and inference mechanisms. The model is designed for off-line analysis, so we are not concerned with distributed application of the model.

Accuracy and correlated errors

Let's begin with a closer look at the independence of our coin-toss contributions in terms of accuracy and correlated errors when two agents have different accuracy probabilities. Suppose agent A_i has an accuracy of $P(A_i)$ and agent A_j has an accuracy of $P(A_j)$, where $P(A_i) \geq P(A_j)$. Figure 1 shows an intuitive pictorial of this two-agent situation. Region x in the fully dependent graphic depicts the probability that both agents are correct ($P(A_j)$) and region z depicts the probability that both agents are incorrect ($1 - P(A_i)$). Region y is the probability that the contributions disagree ($1 - (P(A_j) + (1 - P(A_i)))$). When the agents are fully independent, the probability that they agree is $P(A_i)P(A_j)$, shown in region $\bar{x}$, and where they disagree ($(1 - P(A_i))(1 - P(A_j))$), shown in region $\bar{z}$. Region $\bar{y}$ is the probability that the independent agents disagree. Note that $x \geq \bar{x}$ and $z \geq \bar{z}$.

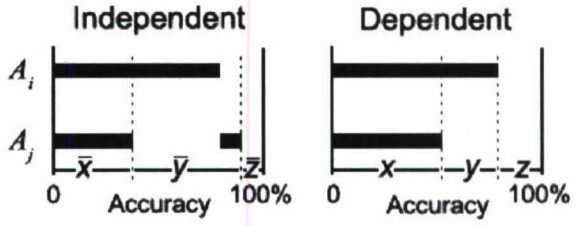


Figure 1: Independence of two contributions. The black bars indicate instances in which the contribution is correct (corresponding instances are aligned vertically.) The instances where A_i is correct are arranged together on the left side of each graph, with instances where A_j is correct also on the left as much as possible within the A_i ordering. The percentage of black versus white is the accuracy of the agent. When the agent contributions are dependent, errors are maximally correlated. Every time the less accurate agent (A_j) is correct, A_i is also correct (region x). Every time the more accurate agent (A_i) is wrong, A_j is also wrong (region z). The remaining region (y) is where their contributions disagree. When the contributions from each agent are independent, A_i and A_j are correlated, but conditionally independent given the coin flip. Region $\bar{x}$ represents where both agents are correct, region $\bar{z}$ where both are wrong, and region $\bar{y}$ is where their contributions disagree.

This pictorial helps in visualizing what occurs as the individual accuracies and the difference between them change. As agent accuracies increase, the contributions naturally become more correlated and the difference between the dependent and independent extremes becomes smaller. At 100% accuracy there is no difference. Therefore, if our agents are close to perfect, incorrect confidence-integration assumptions will have little effect in comparison to mediocre agents operating in the same system.

Given this view of contribution independence, let's look at applying it to integrated contributions.

Concurrent processing model

A set of decisions made by agents is concurrent if the decisions are made without knowledge of any of the other decisions in the set. Figure 2 (a) illustrates the concurrent decisions of three agents, A_1, A_2, and A_3. These agents are all making decisions related to node W, which is the state of the world that the agents are trying to understand. In our model, W can take on any number of values, though for simplicity we will illustrate it for two values, $true$ and $false$. A_1, A_2 and A_3 are all trying to match W's value. This abstraction could represent a coin flip ($W = \{heads, tails\}$) and three agents that are processing images of W and trying

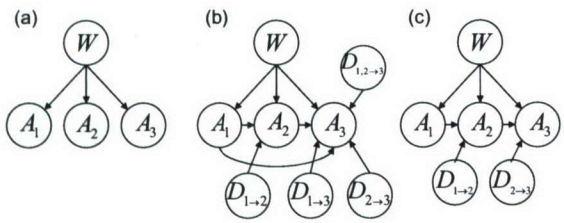


Figure 2: The concurrent processing model. (a) shows nodes representing independent contributions, (b) shows the entire set of dependencies, and (c) shows the subset of dependencies we represent in our model.

to figure out if W was *heads* or *tails*.

In Figure 2 (a) the contributions made by the three agents are independent, while in Figure 2(b) we have represented all potential dependencies between them. The D nodes in Figure 2(b) explicitly represent the error dependencies among the A nodes. However, the full-dependency model quickly becomes unwieldy as the number of agents grows. To keep the number of dependencies manageable, we simplify the model to the linear structure shown in Figure 2(c), where only adjacent nodes are directly dependent on each other and conditional independence is assumed between all non-adjacent nodes. This simplification is reasonable because, at worst, this modeled error will understate the potential error of assuming full contribution independence. If the error is a concern in the simplified 2(c) model, it will be even greater in the full-dependency model.

Use of the D nodes is not necessary, and they could be marginalized out of the model by placing all the dependency information between A_i and A_{i-1} into the link between them. We include an additional node, D_i, for each A_i node (except the first) in our model, because we feel this separation makes the model more intuitive. Each D_i node represents whether the errors in A_i and A_{i-1} are dependent or independent. When $D_i = false$ then A_i is conditionally independent of A_{i-1} given W, $P(A_i|W) = P(A_i|W, A_{i-1})$. When $D_i = true$, however, the errors are maximally correlated with one another (as in the Dependent graphic in Figure 1).

In effect, D_i breaks $P(A_i|A_{i-1}, W)$ down into a mixture of distributions: one distribution for when A_i and A_{i-1} are dependent and one when they are independent, $P(A_i|A_{i-1}, W) = P(D_i = t)P(A_i|A_{i-1}, W, D_i = t) + P(D_i = f)P(A_i|A_{i-1}, W, D_i = f)$. The likelihood that they are dependent is simply the prior probability on D_i. This is similar to the idea of separability [11]. $P(D_i)$ captures the likelihood—not that A_i and A_{i-1} will have the same answer—but that they have the same answer because they are dependent. We can change this prior to represent the range of dependency from conditionally independent ($P(D_i = t) = 0$) to fully dependent ($P(D_i = t) = 1$).

Table 1 shows what this means in terms of the conditional probably table (CPT) for node A_2. All the parameters used in our model are defined in Table 2. With this simple dependency, the directionality of the arrow between A_{i-1} and A_i does not matter. However, given three or more nodes their ordering will make a difference because we are only representing a subset of the possible dependencies (between adjacent nodes) and changing the ordering changes which subset of dependencies we are modeling.

A_1	W	D_2	*true*
t	t	t	$min(\frac{B_2}{B_1}, 1.0)$
f	f	t	$1 - min(\frac{B_2}{B_1}, 1.0)$
t	f	t	$min(\frac{1-B_2}{1-B_1}, 1.0)$
f	t	t	$1 - min(\frac{1-B_2}{1-B_1}, 1.0)$
–	t	f	B_{01}
–	f	f	$1 - B_{01}$

Table 1: The CPT for A_2 showing how the model parameters are used when there are no parents.

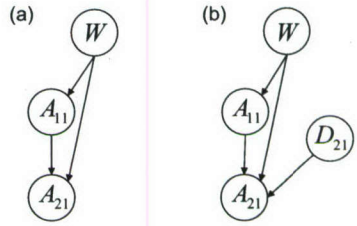


Figure 3: The simplest sequential processing model with one parent node. The result produced by A_{11} is used by A_{21} in its processing. (a) shows the model without an explicit dependence. (b) makes the dependency explicit with the node D_{21}.

Sequential processing model

Two decisions are sequential if the output of one decision is influenced by the output of another decision. While in the concurrent model the dependency between the contributions of two agents could range from conditionally independent to completely dependent, in the sequential model the agents are inherently dependent from the start. For example, in the coin-flip MAS, the *Hair* agent might identify a region in the image that has hair-like texture, while the *Nose* agent hypothesizes a nose. Both of these contributions are sent to the *Head* agent which evaluates those contributions and hypothesizes a region it believes to be someones head on the coin. Hair and nose are correlated with the obverse side of the coin, and identification of a head has an even stronger correlation with the obverse side. The additional processing that the *Head* agent performed using the contributions it received improves the MAS's confidence in the answer over just hair and nose. This kind of sequential processing is represented in Figure 3 for one parent and Figure 4 for two.

Figure 4(a) shows a simple example of sequential processing where contributions of agents A_{11} and A_{12} form the input to agent A_{22}. The integrated contribution A_{22} depends on the output of its parents as well as whether its parents' contributions are correct, hence the link from W to A_{22}. When the parents are both correct there is some chance that the child will introduce an error (parameter I^2_{ij} in Table 2). When the parents are both wrong there is some chance that the child will be able to compensate for the parent errors and determine the correct answer anyway (parameter C^2_{ij} in Table 2). When the parents disagree then the child is left to come up with its own answer (parameter B_{ij} in Table 2). Clearly this is a simplification of the possible relationships between parents and children, but it is sufficient for our analysis model.

A down side to sequential processing comes from detri-

Parameter	Label	Description
W	prior event	Prior probability of event
D	dependency	Prior probability that the nodes are dependent
D^p	parent	Probability that child is dependent on its parent instead of its neighbor to the left, when it is dependent (as given by D)
B_{ij}	base correctness	Probability that child A_{ij} is correct given no parents or parents who disagree
I^1_{ij}	introduce errors	Probability that child is incorrect given its one parent is correct
I^2_{ij}	introduce errors	Probability that child is incorrect given its two parents are correct
C^1_{ij}	correct errors	Probability that child is correct given its one parent is incorrect
C^2_{ij}	correct errors	Probability that child is correct given its two parents are incorrect

Table 2: The parameters used in the CPTs in the general analysis model.

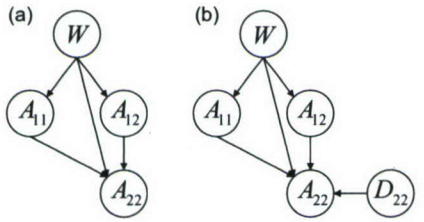


Figure 4: The sequential processing model with two parents. A_{22} takes as input the output of both A_{11} and A_{12}. (a) shows the model without the explicit dependency, (b) shows the model with the explicit dependency D_{22}.

(a) CPT for A_{21}

A_{11}	W	D_{21}	*true*
t	t	f	$1 - I^1_{21}$
f	f	f	I^1_{21}
t	f	f	$1 - C^1_{21}$
f	t	f	C^1_{21}
t	–	t	1.0
f	–	t	0.0

(b) CPT for A_{22}

A_{11}	A_{12}	W	D_{22}	*true*
t	t	t	f	$1 - I^2_{22}$
f	f	f	f	I^2_{22}
t	t	f	f	$1 - C^2_{22}$
f	f	t	f	C^2_{22}
t	f	t	f	B_{22}
f	t	t	f	B_{22}
t	f	f	f	$1 - B_{22}$
f	t	f	f	$1 - B_{22}$
–	t	–	t	1.0
–	f	–	t	0.0

Table 3: The CPTs for nodes A_{21} (Figure 3b) and A_{22} (Figure 4b) in the sequential model.

mental *information cascades* [1]. If one agent makes a mistake it can mislead the next agent in the sequence, and this wrong result can cascade, leading the entire sequence to generate incorrect results. While this phenomenon could be captured in the existing links between the parent nodes and the child, we explicitly represent it with a dependency node, D_{ij}. This is similar to the approach that we used with concurrent processing to explicitly represent contribution dependencies (see Figures 3(b) and 4(b)). When D_{ij} is true, A_{ij} takes the same value as $A_{i-1,j}$ instead of depending on both of its parent nodes. Table 3(a) gives the CPT for the node A_{21} in the one parent case, and Table 3(b) for the two parent case. This representation can be extended

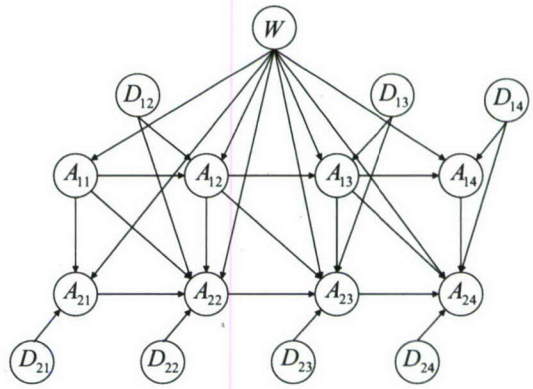


Figure 5: A combined model.

easily to more than two parents. Again, Table 2 provides a description of the parameters used.

Combining both models

The concurrent and sequential processing models can be combined into a general model of an arbitrarily complex system, such as the grid shaped model shown in Figure 5. The rows in the grid represent concurrent processing, while the columns model the sequential processing.

The concurrent and sequential processing models interact in two ways. First, we merge the dependency node D_{ij} for A_{ij} from the two models into one node with three values:

- *parent*—A_{ij} produces the same result as its parent directly above due to information cascading.
- *neighbor*—A_{ij} produces the same result as its neighbor to the left due to dependency.
- *false*—A_{ij} is not forced to answer a particular way due to a dependency.

Changing the priors on the D_{ij} nodes allows the levels of dependence between the nodes to be adjusted easily in order to see how confidence and expected error changes for different degrees of dependence.

The second change introduced by combining the two processing models deals with the accuracy of a node when its parents are dependent. The basic idea is that when A_{ij}'s parents are independent, the rate that it introduces errors or corrects errors will be different then when its parents are dependent. For example, suppose a *Head* agent trying to locate a head on an image of a coin is provided the image

$A_{i,j-1}$	$A_{i-1,j-1}$	$A_{i-1,j}$	W	$D_{i-1,j}$	D_{ij}	*true*
–	t	t	t	p/f	f	$1 - I^2_{ij}$
–	f	f	f	p/f	f	I^2_{ij}
–	t	t	f	p/f	f	$1 - C^2_{ij}$
–	f	f	t	p/f	f	C^2_{ij}
–	t	f	t	p/f	f	B_{ij}
–	f	t	t	p/f	f	B_{ij}
–	t	f	f	p/f	f	$1 - B_{ij}$
–	f	t	f	p/f	f	$1 - B_{ij}$
–	t	t	t	n	f	$1 - I^1_{ij}$
–	f	f	f	n	f	I^1_{ij}
–	t	t	f	n	f	$1 - C^1_{ij}$
–	f	f	t	n	f	C^1_{ij}
–	t	f	t	n	f	B_{ij}
–	f	t	t	n	f	B_{ij}
–	t	f	f	n	f	$1 - B_{ij}$
–	f	t	f	n	f	$1 - B_{ij}$
t	–	–	t	–	n	$min(\frac{B_{ij}}{B_{i,j-1}}, 1.0)$
f	–	–	f	–	n	$1 - min(\frac{B_{ij}}{B_{i,j-1}}, 1.0)$
t	–	–	f	–	n	$min(\frac{1-B_{ij}}{1-B_{i,j-1}}, 1.0)$
f	–	–	t	–	n	$1 - min(\frac{1-B_{ij}}{1-B_{i,j-1}}, 1.0)$
–	–	t	–	–	p	1.0
–	–	f	–	–	p	0.0

Table 4: The CPT for A_{ij}. This illustrates the interaction between the sequential dependency and the concurrent dependency. The D nodes have three values: n for ***neighbor***, p for ***parent***, and f for ***false***.

of the coin and contributions from two *Hair* agents containing the regions of the image that they each believes to be hair. It would be reasonable to expect that if the regions in the contributions are in agreement, and they were generated independently, then the *Head* agent would be less likely to make a mistake than if the *Hair* agents agreed simply because some factor in the environment forced them to agree (such as excessive wear on the coin).

We handle this by adding a link from the dependency node of the relevant parent to the child. When that dependency is *parent* or *false* we have the same probabilities as in Table 3(b). However, when the dependency is *neighbor*, we use different parameters I^1_{ij} and C^1_{ij} instead of I^2_{ij} and C^2_{ij}. See Table 4 for the entire CPT.

3. ANALYSES USING THE MODEL

The combined model can be used to to measure the sensitivity of complex MASs with both concurrent and sequential processing to incorrect confidence-integration assumptions. In this section we demonstrate using the model by analyzing a family of hypothetical systems in which the model nodes are arranged in a rectangular grid shape. Many specific systems would often result in other shape models, based on their contribution interactions. For example, an hourglass shape could reveal the effects of confidence-integration in a system with information bottlenecks. However, by varying the width, depth, and other parameters of our hypothetical grid-system family, we can easily explore some general characteristics of confidence integration.

We make several assumptions about the contribution-exchange structure of our hypothetical systems. For example, there are two parent nodes for the child nodes in each step of sequential processing while a specific system might have some nodes with many parents and some with as few as one. Another assumption is that our agent contributions are equally likely ($P(A = t|W = t) = P(A = f|W = f)$).

After the MAS determines its answer, we are interested in computing the probability that the answer is correct. In this analysis, we assume that we can identify the system's "answer" by observing the values of each A_{ij} in the model. Since the answer the system gives should be consistent with the value having the highest probability, we do not need to explicitly represent it in our model. We assume the system answer is the value of W that maximizes $P(W|A)$ where A is the set of all A_{ij}. Our confidence is then $max_W P(W|A)$. If we were modeling a specific system instance, we might be more interested in values at one or more specific nodes.

This is how we compute the confidence of a particular problem instance in our grid systems, but we want to evaluate the sensitivity of assuming the nodes are independent, $P(D) = 0$, when in fact they are not, $P(D) > 0$. First we can measure our **expected belief** that our answer is correct by taking a weighted average over all possible instances of the observed data:

$$\sum_A P(A) \cdot max_W P(W|A)$$

If agents assume that $P(D) = 0$, then our expected belief is:

$$\sum_A P(A) \cdot max_W P(W|A, P(D) = 0)$$

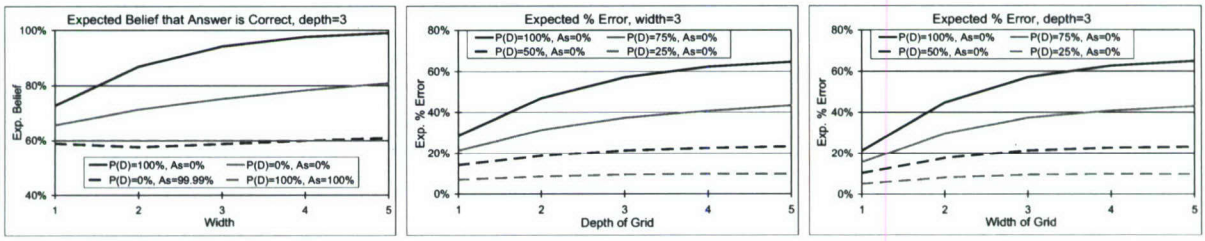
Figure 6(a) illustrates the expected belief for four different cases, two where the independence assumption is correct and two where it is incorrect. If we examine the two lines where $P(D) = 100\%$, the amount of error involved in assuming that $P(D) = 0\%$ is roughly the difference between them. We compute the actual error as a percentage change in the belief:

$$Error(A) = \left| \frac{P(W|A) - P(W|A, P(D) = 0)}{P(W|A)} \right|$$

As before, we can compute the **expected error** by doing a weighted average, $\sum_A P(A) \cdot Error(A)$. Figures 6(b) and 6(c) show the expected percent error for two different shaped grid models. What we see is that even when the dependency between the nodes is fairly small, 25%, we see an expected error in our belief of 10%. As the dependency grows, the percent error can become huge.

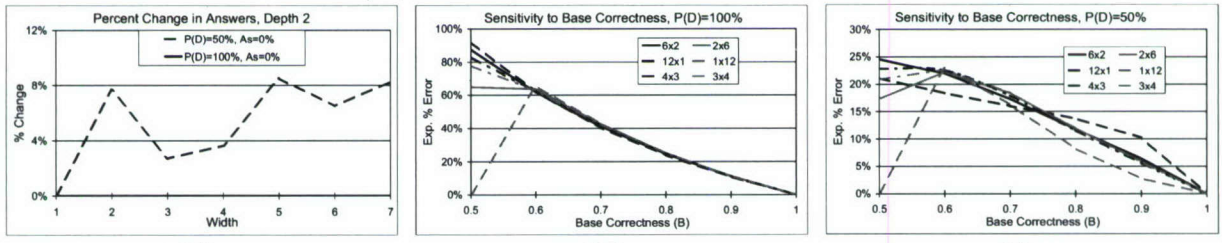
For those applications in which the confidence in the answer is less important than the actual answer, we show in Figure 7(a) the percentage of the answers that are different when assuming independence versus dependence. This does not necessarily mean that the answer is incorrect, just that if the level of dependence had been correctly modeled then the best answer would have been different. In 2xn systems this is trending towards 8% of the time.

We next explore the sensitivity of our family of systems to significant changes in accuracy of contributions. As one would expect, as a system becomes more accurate, the room for independence-assumption error decreases—even when there is high dependency. The flip side is also true. If the



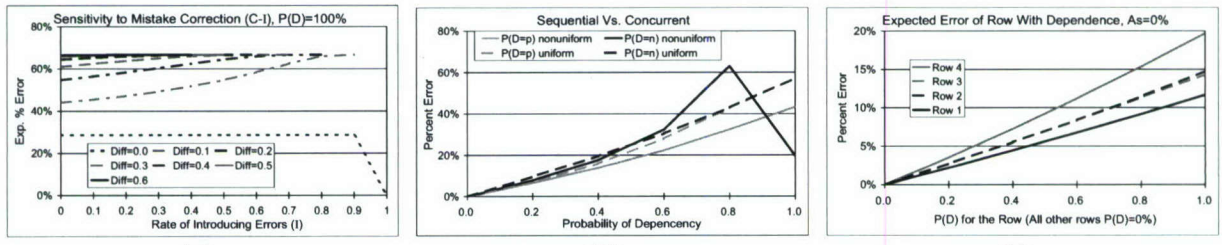
(a) (b) (c)

Figure 6: Expected belief and percent error for grid models of different width and depth. $P(D)$ is the actual dependence between the nodes, while As is the assumption made about the level of independence. $P(D) = 100\%$ means that the nodes are completely dependent, and an $As = 0\%$ means that agents are assuming that $P(D) = 0\%$. (a) The expected belief that the answer the system gives is correct, which is also the expected confidence. (b) The expected percent error as depth increases. (c) The expected percent error as width increases. The parameters used for these charts are $W = 0.5$, $D^p = 0.5$, $B = 0.6$, $I^1 = 0.2$, $I^2 = 0.1$, $C^1 = 0.4$, $C^2 = 0.3$.



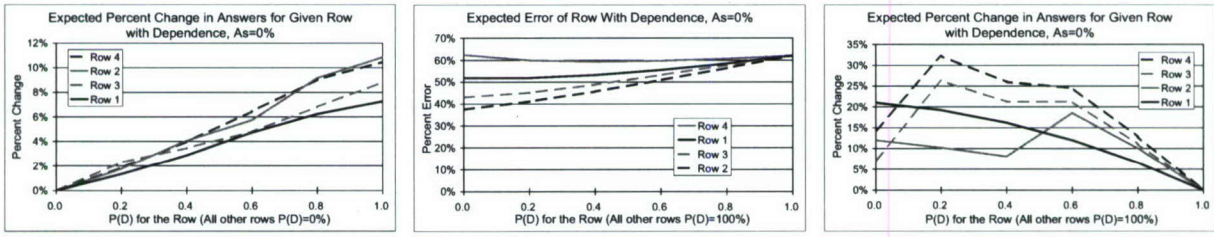
(a) (b) (c)

Figure 7: Effect of independence assumption on system answer and expected error sensitivity to agent accuracy. (a) The percentage of the time that the answer given when assuming independence differs from the answer given if the level of dependence is correctly modeled. When the dependence is 100%, the answer never changes because the nodes can only either be all $true$ or all $false$. (b) The sensitivity of the expected error to the base accuracy (B) of the nodes, when $P(D) = 100\%$. Each line represents a graph with the same number of nodes, but a different shape. The shape only has an effect when the accuracy of the nodes is near random ($B = 50\%$). (c) The sensitivity of the expected error to the base accuracy (B) of the nodes, when $P(D) = 50\%$. In this case the shape of the grid has a small effect on the expected error, but the general trend of each line is the same. The parameters used for these charts are $W = 0.5$, $D^p = 0.5$, $B = 0.6$, $I^1 = 0.2$, $I^2 = 0.1$, $C^1 = 0.4$, $C^2 = 0.3$.



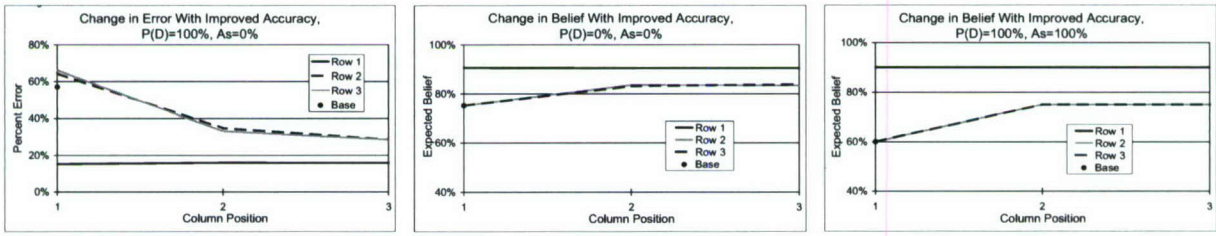
(a) (b) (c)

Figure 8: The effect of error-correction rates, concurrent and sequential dependencies, and row independence. (a) The sensitivity of the expected error to the net rate at which mistakes are corrected, $C - I$. The larger the difference between the rate at which errors are corrected and the rate at which new errors are introduced the less effect the rate of error introduction has on the expected error. In general the effect ranges from high expected error, to higher expected error. (b) This graph compares the difference between the sequential and concurrent dependencies in a 3x3 system. The lines labeled $P(D = p)$ have the probability that the nodes are dependent on its parents (sequential dependence) range from 0 to 1, while the probability that the nodes are dependent on its neighbors (concurrent dependence) is always 0. The lines labeled $P(D = n)$ is the opposite. The lines that are ***uniform*** mean that the base accuracy of the nodes B is the same for all of the nodes, $B = 0.6$, while for the ***nonuniform*** lines the base accuracy of the nodes is 0.6 for the first column, 0.65 for the second, and 0.7 for the third. (c) The expected error when all of the nodes are independent except for one row. The system is most sensitive to having dependency in the last row since there is not any processing after that to correct the mistakes made. The parameters used for these charts are $W = 0.5$, $D^p = 0.5$, $B = 0.6$, $I^1 = 0.2$, $I^2 = 0.1$, $C^1 = 0.4$, $C^2 = 0.3$.



(a) (b) (c)

Figure 9: The effect of single-row dependencies. (a) The expected percentage of answers that change when there is dependence only in the given row. (b) The opposite of Figure 8(c) in that each row is completely dependent except for the given row. (c) The expected percentage of answers that change when there is complete dependence in all rows but the given row. The parameters used for these charts are $W = 0.5$, $D^p = 0.5$, $B = 0.6$, $I^1 = 0.2$, $I^2 = 0.1$, $C^1 = 0.4$, $C^2 = 0.3$.



(a) (b) (c)

Figure 10: Effect of single-node accuracy. These figures were obtained by taking a 3x3 grid system and changing the base accuracy B from 0.6 to 0.9 for one node only. The node is specified by the row given by the line and the column given by the x-axis. The dot labeled ***base*** is the value obtained when no node's accuracy is increased. (a) is the expected percent error. (b) The change in expected belief for increasing the accuracy of one node when the nodes are independent and agents (correctly) assume that they are independent. (c) This is the change in expected belief when the nodes are completely dependent and agents (correctly) assume that they are dependent. The parameters used for these charts are $W = 0.5$, $D^p = 0.5$, $B = 0.6$, $I^1 = 0.2$, $I^2 = 0.1$, $C^1 = 0.4$, $C^2 = 0.3$.

system's decisions are mostly random, then even when there is strong dependence it cannot get much worse by assuming independence. Figures 7(b) and 7(c) illustrate this by varying the base accuracy parameter from random, $B = 0.5$, to perfect, $B = 1.0$.

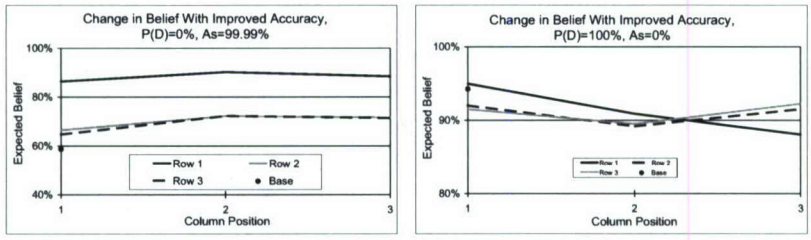
Figure 8(a) shows that the percent error is not very sensitive to the rate that mistakes are corrected versus introduced in sequential processing. Figure 8(b) shows the difference effect on percent error between sequential and concurrent dependencies, and Figure 8(c) shows the effect on percent error when all nodes are independent except for one selected row in the grid. As one might expect, later rows in sequential processing had a greater effect on percent error.

Figure 9(a) explores the effect on changed answers when all nodes are independent except for one selected row in the grid. Figure 9(c) shows the opposite case, where only the given row is independent. The percent error in this case is shown in Figure 9(b) (the opposite case of the percent error shown in Figure 8(c)).

The next figures explore the effect of an accuracy change to one node (agent) in the system. Figure10(a) shows the resulting change to percent error. The error when all of the nodes have an accuracy of 0.6 is 57%. Increasing the accuracy of nodes 2,1 and 3,1 in column 1 of our model actually increases the percent error value slightly. This is an edge effect of our modeling simplifications stemming from the fact that, with one parent, those two nodes in the model do not include the base correctness B in their CPTs. A more accurate measure for percent-error effects stemming from column one can be obtained by using a second model in which columns are exchanged. Figure 10(b) shows the change in expected belief for increasing the accuracy of one node when the nodes are independent and agents (correctly) assume that they are independent, while Figure 10(c) shows the change in expected belief when the nodes are completely dependent and agents (correctly) assume that they are dependent. Figures 11(a) and 11(b) shows the effect of one node's accuracy on the change in expected belief under incorrect dependency assumptions. Figures 12(a) and 12(b) show the change in system answer.

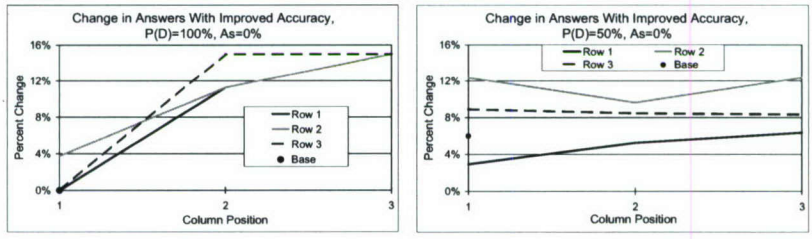
4. CONCLUSION

In this paper we demonstrated that a domain-independent model can be used to analyze the effect and propagation of potentially incorrect confidence-integration assumptions in a collaborating MAS application. We developed representations for both concurrent and sequential contribution processing and explored the implications of unaccounted-for dependencies in a hypothetical family of applications (modeled as rectangular grids) operating under a range of accuracy and dependency conditions. We showed that, although incorrectly assuming contribution independence is tolerable when agents are close to perfect, incorrect independence assumptions can be significant in systems involving mediocre agents. This can result in inaccurate result confidence values and changed system answers.



(a) (b)

Figure 11: Single-node accuracy and change in expected belief due to incorrect dependency assumptions. (a) is the change when the nodes are independent and we incorrectly assume that they are dependent. (b) is the change when the nodes are dependent and we incorrectly assume that they are independent. The parameters used for these charts are $W = 0.5$, $D^p = 0.5$, $B = 0.6$, $I^1 = 0.2$, $I^2 = 0.1$, $C^1 = 0.4$, $C^2 = 0.3$.



(a) (b)

Figure 12: Single-node accuracy and change of answer due to incorrect dependency assumptions. In (a) the nodes are completely dependent and agents assume they are independent. In (b) the nodes are 50% dependent and agents assume that they are independent. The parameters used for these charts are $W = 0.5$, $D^p = 0.5$, $B = 0.6$, $I^1 = 0.2$, $I^2 = 0.1$, $C^1 = 0.4$, $C^2 = 0.3$.

Our modeling approach can be used to help in the design and improvement of MASs. If contributions are redundant, work by agents to develop them can be eliminated (unless the redundancy is desired due to possible agent failures). If additional data or more capabable processing can improve the accuracy of agent decisions, our model can identify the agents where decision-accuracy improvement efforts will produce the most overall system benefit.

An important future direction is to design a domain-independent mechanism that can be used to efficiently represent the dependence among contributions so that an independence assumption is unnecessary. The challenge is two-fold. First, the number of potential dependencies to capture grows exponentially with the number of entities in the system. Second, multi-agent and blackboard systems are typically open, and entities can come and go. While it is reasonable to measure or estimate the accuracy of decisions made by individual entities, assuming we know all of the dependencies among them is unrealistic. Entities may be added later while the system is already running, and some of them may not even exist when the application is developed. One possible approach is to characterize the error probabilities under specific conditions for each entity and to use these characterizations to estimate tighter bounds on the situational dependence/independence between entity contributions.

5. REFERENCES

[1] S. Bikhchandani, D. Hirshleifer, and I. Welch. A theory of fads, fashion, custom, and cultural change as informational cascades. *Journal of Political Economy*, 100(5):992–1026, Oct. 1992.

[2] J. M. Bradshaw. *Software Agents.* AAAI Press & MIT Press, 1997.

[3] D. D. Corkill. Blackboard systems. *AI Expert*, 6(9):40–47, Sept. 1991.

[4] D. D. Corkill. Collaborating software: Blackboard and multi-agent systems & the future. In *Proceedings of the International Lisp Conference*, New York, New York, Oct. 2003.

[5] R. S. Engelmore and A. Morgan, editors. *Blackboard Systems.* Addison-Wesley, 1988.

[6] J. Ferber. *Multi-Agent Systems: Toward a collective intelligence.* Addison-Wesley, 1998.

[7] B. J. Grosz. Collaborative systems. *AI Magazine*, 17(2):67–85, Summer 1996.

[8] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt. An integrated trust and reputation model for open multi-agent systems. *Journal of Autonomous Agents and Multi-Agent Systems*, 13(2):119–154, Mar. 2006.

[9] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

[10] J. Pearl. *Causality: Models, Reasoning and Inference.* Cambridge University Press, 2000.

[11] A. Pfeffer. Sufficiency, separability and temporal probabilistic models. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*, pages 421–428, Seattle, Washington, Aug. 2001.

[12] J. Surowiecki. *The Wisdom of Crowds.* Doubleday, 2004.